User Manual

# Anybus® X-gateway CANopen EtherNet/IP

**Doc.Id. HMSI-168-85**
**Rev. 2.01**

Connecting Devices™

*HALMSTAD • CHICAGO • KARLSRUHE • TOKYO • BEIJING • MILANO • MULHOUSE • COVENTRY • PUNE • COPENHAGEN*

HMS Industrial Networks
Mailing address: Box 4126, 300 04 Halmstad, Sweden
Visiting address: Stationsgatan 37, Halmstad, Sweden

E-mail: info@hms-networks.com
Web: www.anybus.com

# Important User Information

This document is intended to provide a good understanding of the functionality offered by the Anybus X-gateway CANopen - EtherNet/IP.

The reader of this document is expected to be familiar with high level software design, and communication systems in general. The use of advanced CANopen specific functionality may require in-depth knowledge in CANopen networking internals and/or information from the official CANopen specifications. In such cases, the people responsible for the implementation of this product should either obtain the CANopen specification to gain sufficient knowledge or limit their implementation in such a way that this is not necessary. Also knowledge of EtherNet/IP is expected.

## Liability

Every care has been taken in the preparation of this manual. Please inform HMS Industrial Networks AB of any inaccuracies or omissions. The data and illustrations found in this document are not binding. We, HMS Industrial Networks AB, reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be considered as a commitment by HMS Industrial Networks AB. HMS Industrial Networks AB assumes no responsibility for any errors that may appear in this document.

There are many applications of this product. Those responsible for the use of this device must ensure that all the necessary steps have been taken to verify that the applications meet all performance and safety requirements including any applicable laws, regulations, codes, and standards.

HMS Industrial Networks AB will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features, timing, or functional side effects found outside the documented scope of this product. The effects caused by any direct or indirect use of such aspects of the product are undefined, and may include e.g. compatibility issues and stability issues.

The examples and illustrations in this document are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular implementation, HMS Industrial Networks AB cannot assume responsibility for actual use based on these examples and illustrations.

## Intellectual Property Rights

HMS Industrial Networks AB has intellectual property rights relating to technology embodied in the product described in this document. These intellectual property rights may include patents and pending patent applications in the US and other countries.

## Trademark Acknowledgements

Anybus ® is a registered trademark of HMS Industrial Networks AB. All other trademarks are the property of their respective holders.

| | |
|---|---|
| **WARNING**: | This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures. |
| **ESD Note**: | This product contains ESD (Electrostatic Discharge) sensitive parts that may be damaged if ESD control procedures are not followed. Static control precautions are required when handling the product. Failure to observe this may cause damage to the product. |
| **WARNING**: | DO NOT REMOVE OR REPLACE USB CONNECTOR WHILE CIRCUIT IS LIVE UNLESS THE AREA IS KNOWN TO BE FREE OF IGNITIBLE CONCENTRATIONS OF FLAMMABLE GASES OR VAPORS. |


Copyright© HMS Industrial Networks AB
Jan 2013 Doc Id HMSI-168-85

# Table of Contents

**Preface About This Document**

Related Documents ..... 1
Document History ..... 1
Conventions & Terminology ..... 2
Sales and Support ..... 3

**Chapter 1 Anybus X-gateway CANopen - EtherNet/IP**

Introduction ..... 4
Features ..... 5
Functional Overview ..... 6
Data Exchange ..... 6

**Chapter 2 About the Anybus X-gateway CANopen**

External View ..... 8
Status LEDs ..... 9
Primary Network ..... 10
*Ethernet Port Connectors* ..... 10
Secondary Network ..... 11
*CANopen Connector* ..... 11
*Configuration Switches* ..... 11
USB Connector ..... 12
*Power Connector* ..... 12
Hardware Installation ..... 13
CANopen Electronic Data Sheet (EDS) ..... 14
EtherNet/IP Electronic Datasheet (EDS-file) ..... 14

**Chapter 3 Getting Started**

**Chapter 4 CANopen Fieldbus Functionality**

Supported Fieldbus Services ..... 16

**Chapter 5 Configuration**

Module Identification ..... 17
CANopen Master/Slave Configuration ..... 18
Secondary CANopen Network Configuration ..... 19
*LSS Routine* ..... 19
Configuration of the EtherNet/IP interface ..... 20
*Connection parameters* ..... 20
*IP Settings* ..... 20
Enabling Data Exchange ..... 21

## Chapter 6 CANopen Module Specification

NMT State Machine ..... 22
Data Exchange ..... 23
*Control Word* ..... 24
*Status Word* ..... 25
*Example* ..... 27
*PDO Functionality* ..... 28
LSS Services ..... 29
Error Control ..... 30
*Heartbeat Mechanism* ..... 30
*Node Guarding* ..... 30
*Emergency Object (EMCY)* ..... 30

## Chapter 7 CANopen Supported Objects

Static Data Types ..... 31
Communication Profile Area ..... 31
*DS301 Communication Profile Objects* ..... 31
*Configuration Manager* ..... 33
*Network Management Objects* ..... 34
Vendor Specific Objects ..... 39
*Transmit Buffer* ..... 40
*Receive Buffer* ..... 41
*I/O Buffer Addresses and Object Dictionary Indices Relation* ..... 42
*General Fieldbus Parameters* ..... 43
*EtherNet/IP Specific Parameters* ..... 43

## Appendix A Technical Specification

Protective Earth (PE) Requirements ..... 44
Power Supply ..... 44
Environmental Specification ..... 44
*Temperature* ..... 44
*Relative Humidity* ..... 44
EMC (CE) Compliance ..... 45
UL and ATEX Certification ..... 45

## Appendix B Status LED Timing Diagrams

## Appendix C CANopen Emergency Codes

## Appendix D CANopen Configuration Example

Configuration Example ..... 48
Enabling Data Exchange ..... 52

# P. About This Document

For more information, documentation etc., please visit the HMS web site, 'www.anybus.com'.

## P.1 Related Documents

| Document | Author |
|---|---|
| CiA Draft Standard 301 v4.2 | CAN in Automation |
| CiA Draft Standard Proposal 302 Part 1-5 | CAN in Automation |
| EtherNet/IP Specifications Vol 1, 2 | ODVA |
| | |
| | |

## P.2 Document History

### Summary of Recent Changes (2.00 ... 2.01)

| Change | Page(s) |
|---|---|
| Clarified description of status word | 25 |
| | |
| | |
| | |
| | |
| | |

### Revision List

| Revision | Date | Author(s) | Chapter(s) | Description |
|---|---|---|---|---|
| 1.00 | 2010-01-17 | KeL | - | First official release |
| 1.01 | 2011-02-04 | KeL | - | Minor corrections and updates |
| 1.02 | 2011-03-29 | KeL | 6, 7 | Minor corrections |
| 2.00 | 2011-12-01 | KeL | All | General rewrite |
| 2.01 | 2013-01-11 | KeL | 6 | Minor correction |
| | | | | |
| | | | | |
| | | | | |

# P.3 Conventions & Terminology

The following conventions are used throughout this manual:

- Numbered lists provide sequential steps
- Bulleted lists provide information, not procedural steps
- The terms 'Anybus' or 'module' refers to the Anybus X-gateway module.
- Hexadecimal values are written in the format NNNNh, where NNNN is the hexadecimal value.
- A byte always consists of 8 bits

# P.4 Sales and Support

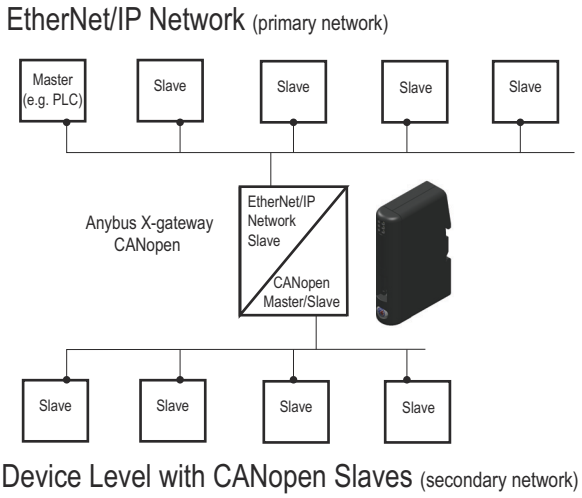
| Sales | | Support | |
|---|---|---|---|
| **HMS Sweden (Head Office)** | | | |
| E-mail: | sales@hms-networks.com | E-mail: | support@hms-networks.com |
| Phone: | +46 (0) 35 - 17 29 56 | Phone: | +46 (0) 35 - 17 29 20 |
| Fax: | +46 (0) 35 - 17 29 09 | Fax: | +46 (0) 35 - 17 29 09 |
| Online: | www.anybus.com | Online: | www.anybus.com |
| **HMS North America** | | | |
| E-mail: | us-sales@hms-networks.com | E-mail: | us-support@hms-networks.com |
| Phone: | +1-312 - 829 - 0601 | Phone: | +1-312-829-0601 |
| Toll Free: | +1-888-8-Anybus | Toll Free: | +1-888-8-Anybus |
| Fax: | +1-312-629-2869 | Fax: | +1-312-629-2869 |
| Online: | www.anybus.com | Online: | www.anybus.com |
| **HMS Germany** | | | |
| E-mail: | ge-sales@hms-networks.com | E-mail: | ge-support@hms-networks.com |
| Phone: | +49 (0) 721-989777-000 | Phone: | +49 (0) 721-989777-000 |
| Fax: | +49 (0) 721-989777-010 | Fax: | +49 (0) 721-989777-010 |
| Online: | www.anybus.de | Online: | www.anybus.de |
| **HMS Japan** | | | |
| E-mail: | jp-sales@hms-networks.com | E-mail: | jp-support@hms-networks.com |
| Phone: | +81 (0) 45-478-5340 | Phone: | +81 (0) 45-478-5340 |
| Fax: | +81 (0) 45-476-0315 | Fax: | +81 (0) 45-476-0315 |
| Online: | www.anybus.jp | Online: | www.anybus.jp |
| **HMS China** | | | |
| E-mail: | cn-sales@hms-networks.com | E-mail: | cn-support@hms-networks.com |
| Phone: | +86 (0) 10-8532-3183 | Phone: | +86 (0) 10-8532-3023 |
| Fax: | +86 (0) 10-8532-3209 | Fax: | +86 (0) 10-8532-3209 |
| Online: | www.anybus.cn | Online: | www.anybus.cn |
| **HMS Italy** | | | |
| E-mail: | it-sales@hms-networks.com | E-mail: | it-support@hms-networks.com |
| Phone: | +39 039 59662 27 | Phone: | +39 039 59662 27 |
| Fax: | +39 039 59662 31 | Fax: | +39 039 59662 31 |
| Online: | www.anybus.it | Online: | www.anybus.it |
| **HMS France** | | | |
| E-mail: | fr-sales@hms-networks.com | E-mail: | fr-support@hms-networks.com |
| Phone: | +33 (0) 3 68 368 034 | Phone: | +33 (0) 3 68 368 033 |
| Fax: | +33 (0) 3 68 368 031 | Fax: | +33 (0) 3 68 368 031 |
| Online: | www.anybus.fr | Online: | www.anybus.fr |
| **HMS UK & Eire** | | | |
| E-mail: | uk-sales@hms-networks.com | E-mail: | support@hms-networks.com |
| Phone: | +44 (0) 1926 405599 | Phone: | +46 (0) 35 - 17 29 20 |
| Fax: | +44 (0) 1926 405522 | Fax: | +46 (0) 35 - 17 29 09 |
| Online: | www.anybus.co.uk | Online: | www.anybus.com |
| **HMS Denmark** | | | |
| E-mail: | dk-sales@hms-networks.com | E-mail: | support@hms-networks.com |
| Phone: | +45 (0) 35 38 29 00 | Phone: | +46 (0) 35 - 17 29 20 |
| Fax: | +46 (0) 35 17 29 09 | Fax: | +46 (0) 35 - 17 29 09 |
| Online: | www.anybus.com | Online: | www.anybus.com |
| **HMS India** | | | |
| E-mail: | in-sales@hms-networks.com | E-mail: | in-support@hms-networks.com |
| Phone: | +91 (0) 20 40111201 | Phone: | +91 (0) 20 40111201 |
| Fax: | +91 (0) 20 40111105 | Fax: | +91 (0) 20 40111105 |
| Online: | www.anybus.com | Online: | www.anybus.com |

# 1. Anybus X-gateway CANopen - EtherNet/IP

## 1.1 Introduction

The Anybus X-gateway CANopen is a series of network gateways, used to provide a seamless connection between a primary fieldbus/Ethernet network and a secondary CANopen subnetwork. The gateway enables the master of the fieldbus/Ethernet network to exchange data to and from the secondary CANopen subnetwork. This makes it possible to integrate CANopen devices into almost any other PLC system and their supported networks.

The gateway is based on patented Anybus technology, a proven industrial communication solution used all over the world by leading manufacturers of industrial automation products. Each module offers CANopen master/slave connectivity to one of these industrial networks: EtherCAT, PROFIBUS DPV1, DeviceNet, ControlNet, CANopen, Modbus RTU, EtherNet/IP, PROFINET IO (both RT and IRT) or Modbus TCP.




No proprietary configuration software is needed, though dedicated configuration tools are required when setting up the actual industrial network communications. Any standard CANopen configuration tool[1] can be used to configure the CANopen interface.

The gateways transmit I/O data transparently between the two networks. I/O data from the primary fieldbus/Ethernet network is written inte CANopen objects that can be mapped into CANopen PDOs or read via CANopen SDOs and vice versa.

The gateway, described in this manual, connects a EtherNet/IP network with a CANopen network. The module acts as a EtherNet/IP adapter/slave on the primary network and can act either as a slave or as a master on the subnetwork, transmitting I/O data between the networks.

The EtherNet/IP adapter/slave interface, connected to the primary network, is configured with a standard device description file (GSD/EDS) and the standard configuration tool of the master of that network. No programming is required.

**IMPORTANT:** *This product acts as a gateway between two industrial networks. One network is a CANopen subnetwork, on which the module either acts as a master or as a slave, depending on configuration. Using the module, this CANopen subnetwork is connected to and can exchange data with another kind of industrial network, e.g. PROFIBUS or EtherNet/IP, connected to the module. To make it easier to distinguish the two networks from each other, the CANopen subnetwork will be called the secondary network throughout the manual. The other network will be called the primary network. In the product that this manual describes, the primary network is* EtherNet/IP *and the secondary network is CANopen.*

1. CANopen Configuration Manager from PORT is available for download at www.anybus.com. This version of the tool from PORT can only be used with products form HMS Industrial Networks. Please contact HMS support for further information, see "Sales and Support" on page 3.

# 1.2 Features

The Anybus CANopen X-gateway acts as an intelligent link between two industrial networks. On the secondary CANopen subnetwork, it can perform either as a master (manager) or as a slave (server), depending on configuration, while it always will act as a slave on the primary fieldbus/Ethernet side. The implementation is based on HMS NP30 network microprocessor and is certified by CAN in Automation (CIA) for full conformance to the CANopen DS 301 v4.0.2 standard.

### CANopen (subnetwork, secondary network)

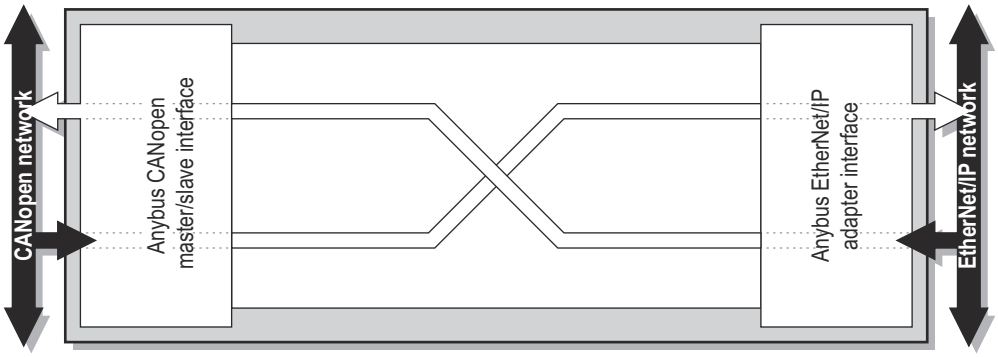
- CANopen master (manager) and slave functionality
- Connects up to 126 CANopen slave nodes
- Complies to the CANopen communication profile DS301 4.2 and DSP302 (part 1-5)
- Supports cyclic and acyclic synchronous as well as COS (change of state) PDO message types
- 20 kbps... 1 Mbps operation
- Heartbeat and node guarding mechanisms
- Sync objects
- 128 receive and 128 transmit PDOs available
- Up to 510 bytes of cyclic data in each direction (PDO)

### EtherNet/IP Features (primary network)

- 10/100 Mbit operation in full or half duplex
- DHCP/Anybus IPconfig (HICP) support
- EtherNet/IP Adapter

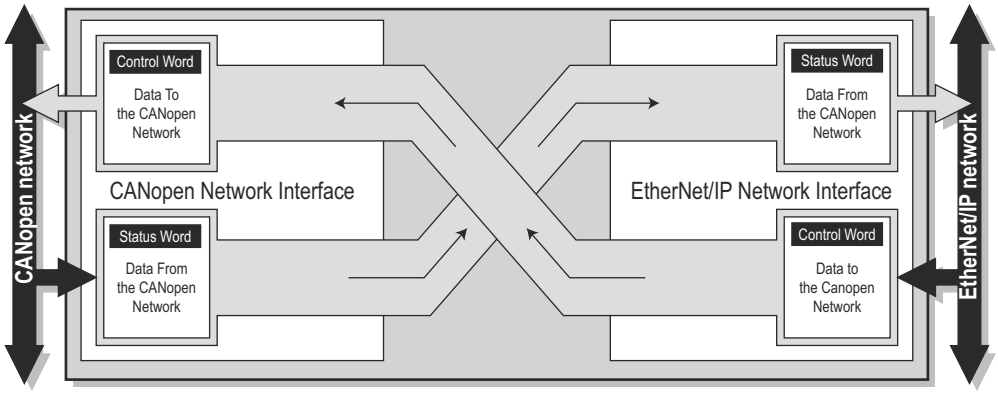
## 1.3 Functional Overview

Internally, the X-gateway consists of an intelligent gateway platform, an Anybus CANopen interface[1] and an Anybus EtherNet/IP interface. The CANopen interface and the Anybus EtherNet/IP interface are interconnected through the intelligent gateway platform, which basically forwards data from one network to the other and vice versa as shown below. This design allows almost any industrial network to be connected to a CANopen master or a slave on a separate CANopen network.




## 1.4 Data Exchange

Each of the two network interfaces exchanges data on its network through two buffers. The gateway forwards the data between these buffers as shown below. Note that this process is separated from the network data exchange. While the gateway ensures data consistency (where applicable), it does not feature any built-in mechanisms for synchronisation between the primary EtherNet/IP network and the secondary CANopen network.




Each buffer holds up to 512 bytes of data, where the first two bytes on the primary network side always are used for control/status information. The remaining 510 bytes gives the theoretical upper limit for the number of data bytes that can be exchanged in each direction. Please note that the actual number of bytes that can be exchanged is highly application and network dependent and can thus be noticably lower than 510 bytes.

Through the dedicated control word, the master on the primary EtherNet/IP network starts/stops the exchange of data on the secondary CANopen network (the subnetwork). It can also reset the gateway if

1. When it is started the first time, the Anybus X-gateway is set as slave on the secondary network. This can be changed during configuration, see "CANopen Configuration Example" on page 48. The module will remember its setting when restarted.

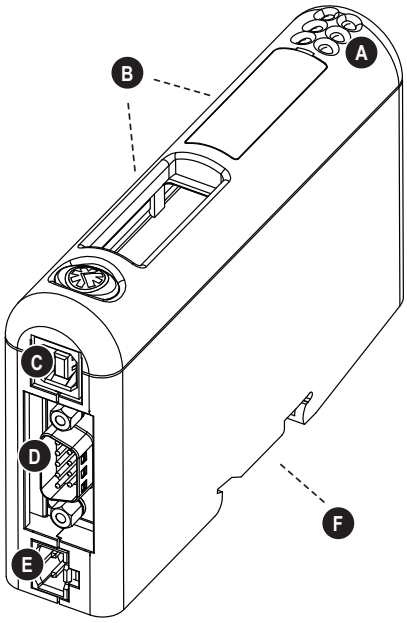
needed. The master on the primary EtherNet/IP network can see the status of the secondary CANopen network in the corresponding status word.

The amount of data that shall be exchanged, and the use of the control- and status functionality, is specified separately for each application. This means that even though up to 510 bytes of data can be forwarded to an interface, the amount of data that will actually be exchanged on the primary EtherNet/IP network is determined by settings of the secondary CANopen network, with consideration taken to the limits of the interface.

The available control- and status functionality is described in "Data Exchange" on page 23. Also note that the terminology and definitions used for different types of data vary greatly between different networking systems. All data transported through the Anybus X-gateway CANopen is fast, cyclic data and is simply referred to as 'I/O Data' in this document.

# 2. About the Anybus X-gateway CANopen

## 2.1 External View




**A: Status LEDs**

See also...

- "Status LEDs" on page 9

**B: Primary Network Connectors**

This connector (connectors) are used to connect the Anybus X-gateway CANopen module to the primary EtherNet/IP network. They are described in "Primary Network" on page 10.

**C: USB connector**

This connector simulates a COM-port, used for software upgrade of the module. Please note that this connector can not be used for configuration of the module.

See also...

- "Secondary Network" on page 11

**D: CANopen Connector**

This connector is used to connect the gateway to the secondary CANopen network.

See also...

- "CANopen Connector" on page 11

**E: Power Connector**

This connector is used to apply power to the gateway.

See also...

- "Power Connector" on page 12

**F: DIN-rail Connector**

The DIN-rail mechanism connects the gateway to PE (Protective Earth).

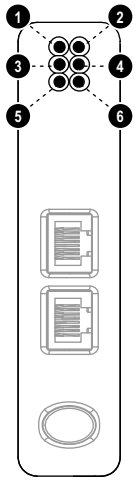
See also...

- "Hardware Installation" on page 13

# 2.2 Status LEDs

The status LEDs on the front indicate the status of the module as shown in the table below. Their behavior is described in "Status LED Timing Diagrams" on page 46

Status LEDs 1 - 4 indicate the status of the primary EtherNet/IP network and status LEDs 5 - 6 indicate the status of the secondary CANopen (sub)network and the device.



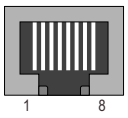

| # | State | Status |
|---|---|---|
| 1 - Module status | Off | No power or initializing |
| | Green | Controlled by a scanner in Run state |
| | Flashing green | Not configured or scanner in Idle state |
| | Flashing red | Minor fault, recoverable |
| | Red | Major fault, unrecoverable |
| | Flashing green/red | Self test in progress |
| 2 - Network Status | Off | No IP address |
| | Green | On-line, one or more connections (CIP class 1 or 3) |
| | Flashing green | On-line, no connection established |
| | Flashing red | One or more connections timed out (CIP class 1) |
| | Red | Duplicate IP address, fatal error |
| | Flashing green/red | Self test in progress |
| 3 - Link/Activity 1<br>4 - Link/Activity 2 | Off | No link sensed on port 1/2 |
| | Flashing green | Activity, receiving/transmitting Ethernet packets at 100 Mbit |
| | Flashing yellow | Activity, receiving/transmitting Ethernet packets at 10 Mbit |
| 5 - CANopen subnet status[a] | Off | Power off |
| | Flickering green/red | The LSS services are in progress |
| | Blinking green | Preoperational state |
| | Single flash, green | Stopped state |
| | On, green | Operational state |
| | Blinking red | Configuration error |
| | Single flash, red | Warning limit reached in CAN controller, e.g. bad or no signal on CANopen network |
| | Double flash, red | Error control event |
| | Triple flash, red | Sync error |
| | Quadruple flash, red | Data communication timeout |
| | Red | Bus off |
| 6 - Device status | Off | Power off |
| | Blinking green | Bootup |
| | On, green | Running |
| | Single flash, red | Initialization error |
| | Double flash, red | Internal timeout |
| | Triple flash, red | Hardware failure |
| | Quadruple flash, red | Invalid switch settings |
| | On, red | Fatal error |

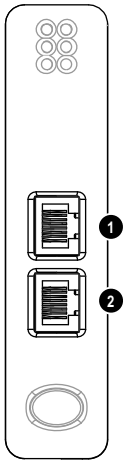
a. This LED shows the status of the secondary CANopen network.

# 2.3 Primary Network

## 2.3.1 Ethernet Port Connectors

| Pin no | Description |
|---|---|
| 1 | TD+ |
| 2 | TD- |
| 3 | RD+ |
| 6 | RD- |
| 4, 5, 7, 8 | (reserved) |

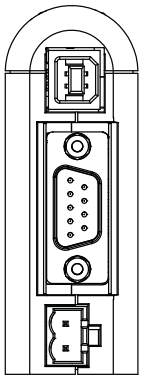


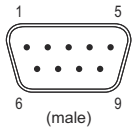
# 2.4 Secondary Network

## 2.4.1 CANopen Connector

At the bottom of the module you find the CANopen connector for the secondary network.



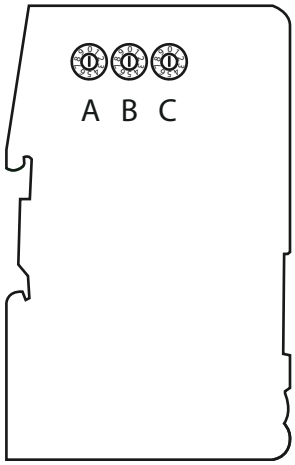
| Pin no. | Description |
|---|---|
| 2 | CAN_L |
| 5 | Housing, CAN cable shield |
| 7 | CAN_H |
| 1, 4, 8, 9 | (reserved) |
| 3, 6 | CAN GND |




This connector is also used to download the CANopen configuration to the module.

## 2.4.2 Configuration Switches

The on-board switches on the side of the module are used to set the CANopen node address and operating baud rate for the interface on the secondary network. These settings cannot be changed during runtime, i.e. the gateway must be restarted in order for any changes to have effect.




**Note:** When these switches have been set, cover them with the switch covers that accompany the module.

### Baud Rate

The baud rate is set via switch A:

| Switch Setting | Baud Rate(kbit/s) |
|---|---|
| 0 | 20 |
| 1 | 50 |
| 2 | 125 |
| 3 | 250 |
| 4 | 500 |
| 5 | 800 |
| 6 | 1000 |
| 7 | Auto[a] |
| 8, 9 | Not available |

a. The automatic baud rate setting should not be used if there is only a small amount of traffic on the bus. This occurs e.g. if the interface is configurated as a CANopen master or if the secondary network is small.

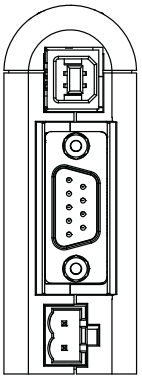
### Node Address

The node address is configured using two rotary switches as follows:

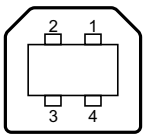
Node Address = (Switch B x 10) + (Switch C x 1)
*Example:* To set node address 42, set switch B to '4' and switch C to '2'.

# 2.5 USB Connector

At the bottom of the module, next to the CANopen connector for the secondary network, you find a USB connector that is only used for software upgrade of the module.



| Pin no. | Description |
|---|---|
| 1 | +5 V input |
| 2 | USBDM (USB communication signals) |
| 3 | USBDP (USB communication signals) |
| 4 | Signal GND |
| Housing | Cable Shield |




This port can only be used for software upgrade.

## 2.5.1 Power Connector

| Pin no. | Description |
|---|---|
| 1 | 24 V DC |
| 2 | GND |

**Notes:**

- Use 60/75 or 75° C copper (CU) wire only.
- The terminal tightening torque must be between 5... 7 lbs-in (0.5... 0.8 Nm)

  See also...

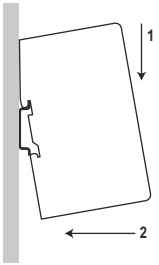
  - "Power Supply" on page 44

# 2.6 Hardware Installation

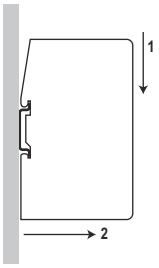
Perform the following steps when mounting the gateway:

1. Set the Node Address and the baud rate for the secondary CANopen network (see "Configuration Switches" on page 11).
2. Snap the gateway on to the DIN-rail (See "External View" on page 8)

   The DIN-rail mechanism works as follows:




To snap the gateway *on*, first press it downwards (1) to compress the spring in the DIN-rail mechanism, then push it against the DIN-rail as to make it snap on (2).




To snap the gateway *off*, push it downwards (1) and pull it out from the DIN-rail (2), as to make it snap off from the DIN-rail.

3. Connect the gateway to the secondary CANopen network.
4. Connect the gateway to the EtherNet/IP network.
5. Connect the power cable and apply power

## 2.7 CANopen Electronic Data Sheet (EDS)

Each device on CANopen is associated with a CANopen Electronic Data Sheet (a.k.a EDS file), which holds a description of the device and its functions. Most importantly, the file describes the object dictionary implementation in the device. This file should be uploaded to the CANopen configuration tool when configuring the secondary CANopen network.

The latest version of the EDS file for the Anybus X-gateway CANopen can be downloaded from the HMS web site, 'www.anybus.com'.

## 2.8 EtherNet/IP Electronic Datasheet (EDS-file)

Each device in a EtherNet/IP network is associated with an Electronic Data Sheet (an EDS file), which describes the implementation of the product. This file is used by the network configuration tool during network configuration.

The latest version of the EDS file for the Anybus X-gateway EtherNet/IP interface can be downloaded from the HMS web site, 'www.anybus.com'.

# 3. Getting Started

The purpose of this chapter is to give a short description on how to install the module and get it up and running, transferring I/O data between the primary EtherNet/IP network and the secondary CANopen subnetwork.

**Note**: The CANopen (secondary) subnetwork interface is configured prior to the primary EtherNet/IP network interface. The Anybus X-gateway CANopen - EtherNet/IP module has to be restarted after this configuration has been finished.

Perform the following steps when installing the gateway:

**1.** Set the CANopen node ID and operating baud rate for the X-gateway on the secondary CANopen network (see "Configuration Switches" on page 11).

**2.** Snap the gateway on to the DIN-rail (See "Hardware Installation" on page 13).

**3.** Connect the gateway to the CANopen (secondary) subnetwork, using the connector at the bottom of the module.

**4.** Connect the power cable and apply power.

**5.** Download the appropriate EDS file from HMS to the external CANopen configuration tool. See "CANopen Electronic Data Sheet (EDS)" on page 14.

**6.** Decide how much data will be transferred. This amount is always configured for the module's interface to the secondary network. A description of how the data is mapped to the Anybus X-gateway CANopen is found in "I/O Buffer Addresses and Object Dictionary Indices Relation" on page 42.

**7.** Consult the "CANopen Configuration Example" on page 48 to configure the module and the secondary CANopen network.

**8.** Connect the gateway to the primary EtherNet/IP network.

**9.** Restart the gateway.

**10.** Download the appropriate EDS file from HMS. See "EtherNet/IP Electronic Datasheet (EDS-file)" on page 14.

**11.** The actual configuration of the module is performed while configuring the EtherNet/IP master/scanner and the EtherNet/IP network, see "Configuration of the EtherNet/IP interface" on page 20. The I/O buffers of the module contains 512 bytes, but the amount of I/O data is decided by the CANopen configuration.

# 4. CANopen Fieldbus Functionality

The functionality of the Anybus X-gateway CANopen master/slave on the secondary network is defined by the CANopen DS301 Rev. 4.2 specification and DSP302 (part 1-5).

**Note**: The first time the module starts up, it starts as a slave on the secondary CANopen network. It can be set as master during configuration, see "CANopen Configuration Example" on page 55. This setting can be saved in the module so that it will start as a master the next time.

## 4.1 Supported Fieldbus Services

Communication and parameters in the CANopen protocol are built around objects. There are different services available to communicate with the objects and to perform other CANopen tasks like supervising the network. The following message types and objects are implemented in the Anybus X-gateway CANopen:

- NMT (Network Management)[1] messages configure and initialize the network, as well as monitor the network and handle errors. If the module is configured as a slave, startup is performed by a master on the network.
- CMT (Configuration Manager)[1] messages are used for configuration of CANopen devices. This primarily involves PDO parameters and mapping of information. If the module is configured as a slave, the configuration is performed by a master on the network.
- PDOs (Process Data Objects) are used for I/O communication. There are 128 Receive PDOs and 128 Transmit PDOs implemented in the Anybus X-gateway CANopen that each can transfer up to 8 bytes. Supported PDO message types are COS (Change of state), Cyclic Synchronous and Acyclic Synchronous.[2]
- SDOs (Service Data Objects) use asynchronous data transmission and are used to access objects without mapping them to an I/O (PDO) connection. Access is provided to all CANopen objects in the module and in the network nodes (master mode). The SDO messages are used to configure the module and they can transfer more than 8 bytes, which is the upper limit for a PDO. (Expedited Upload/Download Protocol and Segmented Upload/Download Protocol are supported)
- A SYNC (Synchronization Object) is used for synchronizing PDO communication. A master can be either a producer or a consumer of the synchronization. A slave can only be a consumer.
- The Heartbeat Mechanism helps a device to monitor the status of another node. The module can appear both as heartbeat producer and consumer.
- The Node Guarding Protocol provides active surveillance of a slave by the master. Slaves can be configured to expect a node guarding request from the master.
- An EMCY (Emergency Object) is used for error reporting when a fatal fault has occurred in the module itself or in other monitored/supervised modules.
- LSS (Layer Setting Services)[1]. An LSS master can configure baud rate and node ID of all slaves that support LSS (i.e. the preconfigured baud rate and node ID of a slave can be changed by a master).

1. Only available when the module is configured as master.
2. The data exchange with the EtherNet/IP network is limited to 512 bytes, affecting the total number of PDOs that can be used in an application.

# 5. Configuration

This chapter describes the configuration of the secondary CANopen network interface as well as the configuration of the primary EtherNet/IP network adapter/slave interface of the module. The secondary CANopen network interface is configured prior to the primary EtherNet/IP network adapter/slave interface. The I/O data sizes configured for the secondary CANopen network decides the data sizes on the primary EtherNet/IP network adapter/slave interface.

## 5.1 Module Identification

The Anybus X-gateway CANopen to EtherNet/IP module identifies itself on the network as follows:

| Description | Value |
|---|---|
| Vendor Code | 90 |
| Vendor Name | “HMS Networks” |
| Product Code | 50 |
| Product Type | 12 |
| Product Type String | “Communications Adapter” |
| Product Name | “Anybus X-gateway CANopen” |
| Catalog | “Anybus X-gateway CANopen” |
| Desc Text | “Anybus X-gateway CANopen” |

# 5.2 CANopen Master/Slave Configuration

The nodes on the secondary CANopen network, including the interface of the Anybus X-gateway on this network, have to be configured using an external CANopen configuration tool[1] running on a computer. The configuration is downloaded to the secondary CANopen network master using a CANopen adapter.[2]

The module is by default configured as a slave at startup. To enable it to perform as a CANopen master, please set this during configuration, see "CANopen Configuration Example" on page 48 and "NMT Start-up, 1F80h" on page 35.

The following parameters have to be defined:

| Parameter | Description | Values | Comment |
|---|---|---|---|
| Node Number | Node ID on the secondary CANopen Network.<br>Allowed values are 1 - 127. | 1-127 | Node ID (1-99) is set using rotary switches, see "Configuration Switches" on page 11. Node IDs above 99 can be set using the configuration tool or from the CANopen network. |
| Baud rate | This parameter defines the baud rate on the secondary CANopen network. If "Auto" is selected the baud rate will be automatically detected. | 20 kbit/s<br>50 kbit/s<br>125 kbit/s<br>250 kbit/s<br>500 kbit/s<br>800 kbit/s<br>1000 kbit/s<br>Auto | Set using rotary switches, see "Configuration Switches" on page 11<br>Automatic baud rate should only be used if the module is used as a slave on the secondary network. |
| Set master | Set the module to be CANopen master. At startup the module is a slave by default. This can be changed during configuration. | | See "NMT Start-up, 1F80h" on page 35. |
| CANopen input data size | This parameter defines the size of the readable data array on the primary network interface. Data to the primary network from the secondary CANopen network. | 2 - 512[a] | The first 2 bytes in the array are used for the status word, so the maximum data size from the secondary CANopen network is 2 bytes less than the maximum value allowed for this parameter. See object 3000h on page 43. |
| CANopen output data size | This parameter defines the size of the writable data array on the primary network interface. Data from the primary network interface to the secondary CANopen network. | 2 - 512[a] | The first 2 bytes in the array are used for the control word, so the maximum data size to the secondary CANopen network is 2 bytes less than the maximum value allowed for this parameter. See object 3001h on page 43. |

a. The data buffers in the Anybus X-gateway CANopen module can hold 512 bytes of data, but the actual maximum data size is highly dependent on network. Please consult the section on configuration of the controlling network further on in this chapter.

**Note**: The input and output data sizes are configured for the secondary CANopen network interface. The master of the primary network will have to take these values into consideration, as they will be used by the primary EtherNet/IP network adapter/slave interface of the module.

1. CANopen Configuration Manager from PORT is available for download at www.anybus.com. This version of the tool can only be used for products from HMS Industrial Networks. Please contact HMS support for further information, see "Sales and Support" on page 3.
2. Please contact HMS support for further information, see "Sales and Support" on page 3.

# 5.3 Secondary CANopen Network Configuration

An external CANopen configuration tool[1] is used to configure the nodes on the secondary CANopen network. Each node can be configured locally or a Concise DCF file can be downloaded to the CANopen network master using a CANopen adapter[2]. At the next startup the CANopen master will configure the network, if this function was set in the configuration tool during initial configuration.

The first time the Anybus X-gateway CANopen is started it starts up as a slave. It can be configured as a master, and if it is, it will continue to start up as a master.

**1.** Download the EDS file[3] for the Anybus X-gateway CANopen from www.anybus.com to your PC.

**2.** Prepare EDS files for all other nodes present on the secondary network.

**3.** Open the CANopen configuration tool.

**4.** Upload the EDS files to the configuration tool.

**5.** Add nodes to the CANopen network.

**6.** Configure each node with the necessary parameters.

**7.** Set Input Data Size in object 3000h and Output Data Size in object 3001h, to define the I/O data size between the secondary CANopen network (subnetwork) and the primary EtherNet/IP network (slave interface). Default values are 16 bytes (14 bytes of data exchanged between the networks + 2 bytes Control/Status Word). See "General Fieldbus Parameters" on page 43.

**8.** Download the configuration to the CANopen network as Concise DCF to the master or store the configuration locally in each module's nonvolatile memory.

Please consult the user manual for the configuration tool for details and/or contact HMS support, see "Sales and Support" on page 3. For an example, see "CANopen Configuration Example" on page 48.

To configure the primary EtherNet/IP adapter/slave interface, restart the module, and then configure the EtherNet/IP network using appropriate configuration tools, see below. Please remember that the amount of data that can be exchanged already is decided by the previous configuration of the secondary CANopen network (CANopen objects 3000h and 3001h). The first two bytes of the input data and the output data are always used for status and control information.

## 5.3.1 LSS Routine

If there is a missing slave on the network after the boot timeout (defined in object 1F89h, page 39) the master will initiate the LSS routine. It will send an identify slave request. If one (and only one) slave responds to that message, the master sets the NodeID of that node to the first available NodeID. The master will then send a bootup request to the slave.

1. CANopen Configuration Manager from PORT is available for download at www.anybus.com. This version of the tool can only be used for products from HMS Industrial Networks. Please contact HMS support for further information, see "Sales and Support" on page 3.
2. Please visit www.anybus.com or contact HMS support for further information, see "Sales and Support" on page 3.
3. The EDS file for the Anybus X-gateway CANopen can be downloaded from www.anybus.com.

# 5.4 Configuration of the EtherNet/IP interface

After the configuration of the CANopen master interface has been finalized, the module has to be re-started before the configuration of the EtherNet/IP adapter interface can be started.

The adapter interface is part of a EtherNet/IP network, and will have to be configured within this. This means that the interface must be assigned an IP address, see "IP Settings" on page 20. The configuration tool used when configuring the EtherNet/IP scanner and the EtherNet/IP network will have to be used to configure the adapter interface as well. Please note that the size of the I/O data that can be read from and written to the module is defined when configuring the CANopen subnetwork interface.

There are a number of different configuration tools for EtherNet/IP available on the market. The choice of tool depends on the application and the EtherNet/IP scanner of the network. An EDS file for the adapter interface is available at 'www.anybus.com'.

An application note, describing how to configure an Anybus EtherNet/IP slave interface with RS Logix and RS Networx, is available on the support pages for the Anybus X-gateway CANopen to EtherNet/IP module at 'www.anybus.com'.

## 5.4.1 Connection parameters

I/O data is accessed in input and output assembly instances on the EtherNet/IP network as shown below. These values are set in the EDS file for the EtherNet/IP interface. Usually the configuration tool reads and uses these values, but there are tools where you have to define them manually.

| | **Assembly instance** |
|---|---|
| **Input** | 100 |
| **Output** | 150 |

## 5.4.2 IP Settings

Three different options for assigning an IP address to the EtherNet/IP slave interface of the module is described below.

### DHCP/BootP (Default)

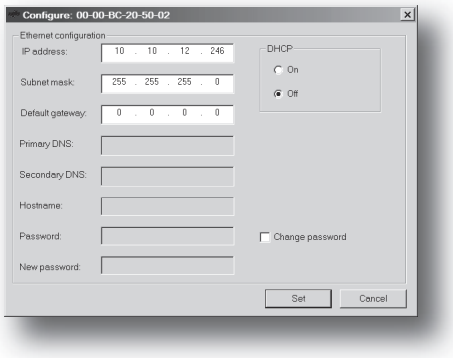
The module retrieves the TCP/IP settings from a DHCP or BootP server.

If it fails and no current settings are available, the module will indicate an error on the on-board status LEDs. The settings may however be configured using HICP, see "Anybus IPconfig (HICP)" on page 21.

### Anybus IPconfig (HICP)

The module supports the HICP protocol used by the Anybus IPconfig utility from HMS, which can be downloaded free of charge from the HMS web site. This utility may be used to configure the network settings of any Anybus product connected to the network. Note that if successful, this will replace the settings currently stored.

Upon starting the program, the network is scanned for Anybus products. The network can be rescanned at any time by clicking 'Scan'. In the list of detected devices, the module will appear as 'Anybus X-gateway CANopen master'. To alter its network settings, double-click on its entry in the list.




A window will appear, containing the IP configuration and password settings. Validate the new settings by clicking 'Set', or click 'Cancel' to abort.

Optionally, the configuration may be protected from unauthorized access by a password. To enter a password, click on the 'Change password' checkbox, and enter the password under 'New password'. When protected, any changes in the configuration requires that the user supplies a valid password.

When done, click 'Set'. The adopted configuration will be stored in the ethernet configuration file.

**Note:** The HICP protocol communicates over UDP port 3250.

## 5.5 Enabling Data Exchange

Once both the interfaces of the X-gateway have been properly configured, the PLC (the master) on the primary network will have to explicitly allow the X-gateway to exchange I/O data, for any I/O data exchange to occur between the primary and secondary networks. To accomplish this, the PLC will write the command "OPERATIONAL" in the control word, see "Control Word" on page 24 for further information.

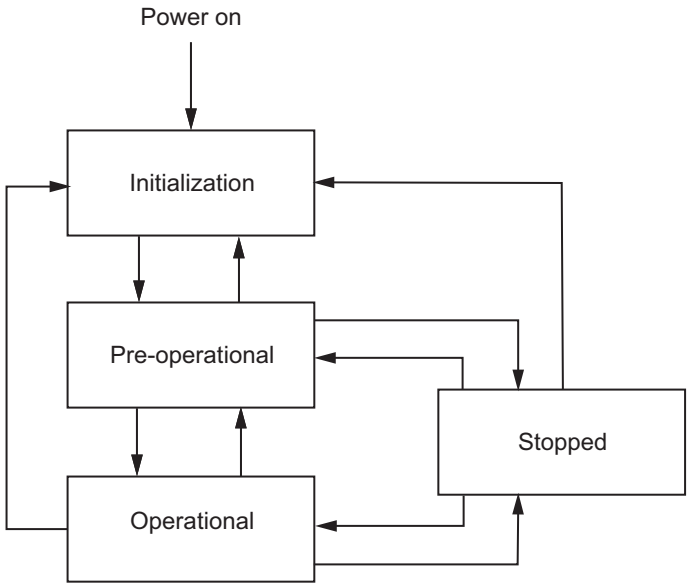
If the module is set as master, it will automatically be available, when the PLC has enabled data exchange. The module will control the secondary network, using the instructions sent in the control world from the PLC.

If the module is set as slave, it will wait for a request form the master of the secondary network before starting to exchange data. If it has not been enabled by the PLC to exchange data, it will return an error message to the secondary network.

# 6. CANopen Module Specification

## 6.1 NMT State Machine

The function of the Anybus X-gateway CANopen can be described as a state machine with four states.




| State | Description |
|---|---|
| Initialization | When the power is switched on, the module starts initializing.<br>The parameters are set to the so called power-on values, which are the default values or the latest stored values. If parameter values are stored from a previous configuration, these are used. If not, or if a restore_default command is issued, the parameters are reset to the default values according to the communication and device profile specifications. |
| Pre-operational | Once initialized, the module automatically enters the pre-operational state. The Anybus X-gateway CANopen can be configured, but no I/O data can be exchanged. |
| Operational | In the operational state all communication objects are active. I/O data is communicated according to the configuration made. |
| Stopped | All communication is stopped, except node guarding and heartbeat, if active. From this state any transition to another state is possible, depending on if a restart, reconfiguration or reset of the module is wanted. |

The module changes states upon reception of a request from an NMT object, a hardware reset or Module Control Services locally initiated by application events.

# 6.2 Data Exchange

The Anybus X-gateway CANopen allows for the exchange of 512 bytes of data in each direction between the primary network and the X-gateway. The first two bytes (the first word) are allocated for a Control/Status word, decreasing the size of I/O data for CANopen to 510 bytes. The actual amount of data that can be exchanged is highly network dependent, see the section on configuration of the primary EtherNet/IP network in chapter 5.

The control and status words of the module are used by the master of the primary EtherNet/IP network to control the Anybus X-gateway CANopen and the secondary CANopen network, and to report the status back from this network. The rest of the I/O data area is available in the CANopen vendor specific object area for real-time data transfer using PDOs (Process Data Objects).

**Note**: The functionality of the Control/Status word differs depending on if the Anybus X-gateway CANopen interface is configured as a slave or as a master on the secondary network.

## 6.2.1 Control Word

The control word is used to control the CANopen network of the Anybus X-gateway CANopen. It is triggered on a CoS (Change of State) event.

| Control Word | | | | Effective I/O Data |
|---|---|---|---|---|
| Byte 0 | | | Byte 1 | Byte 2 - 510 |
| Toggle bit[a] | Cmd, 3 bits | CmdExt, 4 bits | NodeID[b] | Data |

a. The most significant bit in byte 0 is a toggle bit, that is toggled by the controlling network each time a new command is issued.
b. If NodeID = 0, the command is valid only for the node that the module constitutes. If NodeID = 128 (80h), the command is valid for the complete secondary CANopen network. Any other NodeID value will specify the single node that the command is valid for. If the Anybus X-gateway CANopen interface is configured as a slave on the secondary network the only allowed value of NodeID is 0.

### Supported commands

The table below shows available commands and their representation in byte 0 of the control word.

| Toggle bit[a] | Cmd (3 bits) | CmdExt (4 bits) | Name | Master functionality | Slave functionality |
|---|---|---|---|---|---|
| - | 0h | - | (Set NMT State) | This command sets the NMT state of a CANopen node or the CANopen network, according to the value of NodeID. | The NMT state is set by the controlling PLC. If the PLC is running, the NMT state is set to OPERATIONAL. If the PLC is not running, the NMT state is set to PRE-OPERATIONAL.[b] Default: PRE-OPERATIONAL |
| | | 0h | PRE-OPERATIONAL | | |
| | | 1h | OPERATIONAL | | |
| | | 2h | RESET NODE | | |
| | | 3h | RESET COMMUNICATION | | |
| | | 4h | STOP | | |
| | | 5h - Fh | - | (reserved) | |
| | 1h | | Get Node state | This command requests the state set in object 1F82h, see 37, of the CANopen node or network (depending on the value of NodeID).[c d] | |
| | 2h | | Get COPM general status | This command requests the general status of the CANopen module | |
| | 3h - 6h | (reserved) | | | |
| | 7h | - | (No operation) | No Operation<br>It is recommended to set Cmd to this value, when the module goes offline from the fieldbus. This prevents unwanted behavior of the CANopen network, when the fieldbus comes back online. | |

a. The most significant bit in byte 0 is a toggle bit, that is toggled by the controlling network each time a new command is issued.
b. IMPORTANT: The PLC controlling the primary network has to set the X-gateway to OPERATIONAL using the Control Word. If this has not been done, the X-gateway will decline an NMT Set Operational Command on the secondary network by returning an emergency message with the error code FF10h. The same emergency message will be sent when the X-gateway is reset to PRE-OPERATIONAL by the primary network.
c. If the module is configured as slave, only NodeID = 0 is allowed.
d. Only states of nodes monitored by node guarding or heartbeat can be read from object 1F82h.

When started, the module will initialize, and then automatically continue to the state PRE-OPERATIONAL.

I/O data will only be exchanged if the module is in the state OPERATIONAL. To make this possible, the PLC controlling the primary network will have to give the command "Set NMT State (OPERA-

TIONAL)" in the control word. If the module is set as master, it will then administer the secondary network. If the module is set as slave, it will answer to any request from the master of the secondary network to participate in the communication on that network (see footnote in the table above).

The command RESET NODE will restore the module to a previously downloaded configuration. RESET COMMUNICATION will restore the communication settings of the module. In both cases the module will return to the INITIALIZATION state.

*Examples*

| Master/ Slave | Control Word[a] | Meaning |
|---|---|---|
| Slave | 01 00h | 01h: Allow the module to go to OPERATIONAL if asked by an NMT master.<br>00h: The command is only valid for the module itself. |
| Master | 01 80h | 01h Start remote node in the secondary network.<br>80h The command is valid for all nodes in the secondary network. |
| | 01 02h | 01h Start remote node.<br>02h The command is valid for node 2. |
| | 04 80h | 04h Stop remote node.<br>80h The command is valid for all nodes in the secondary network. |

a. The first bit in the control word is toggled for each new command, e.g. changing the first byte from 01h to 81h

## 6.2.2 Status Word

Byte 0 in the status word shows the last valid command and command extension written to the control word, to indicate that the command has been performed. Byte 1 gives the lowest NodeID with error. Please note that there can be one or more nodes, with higher NodeIDs, that also have errors. If NodeID is 0, all nodes are fine. If NodeID is for example 5, it means that there is an error with node 5.

Only errors from nodes monitored by the heartbeat mechanism or by node guarding will be reported. Errors from other slaves can not be recognized.

| Status Word | | | | Effective I/O Data |
|---|---|---|---|---|
| Byte 0 | | | Byte 1 | Byte 2 - 510 |
| Toggle bit[a] | Cmd Rsp, 3 bits | CmdExt, 4 bits | Error Node | Data |

a. The most significant bit in byte 0 is a toggle bit, that is toggled by the module to mirror the toggle bit of the control word.

## Supported commands

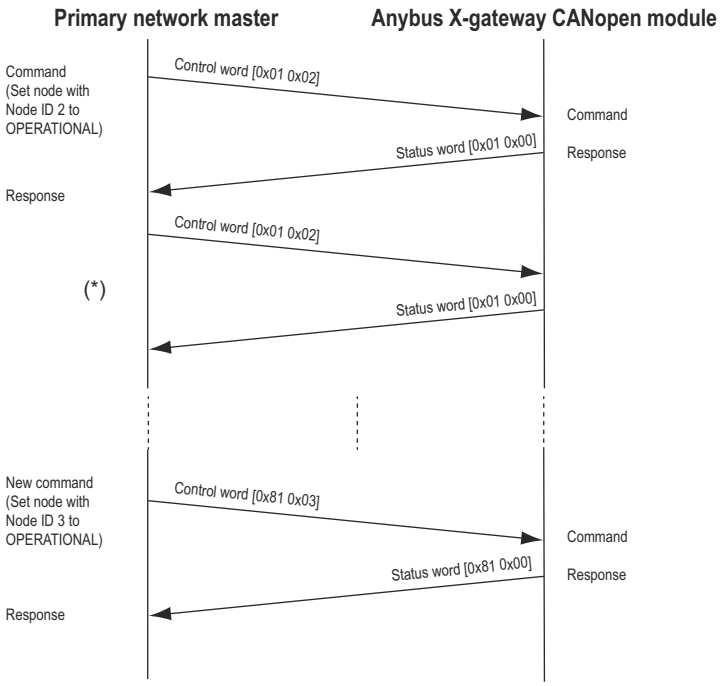
The table below shows available command responses and their representation in byte 0 of the status word.

| Toggle bit[a] | CmdRsp (3 bits) | CmdExtRsp (4 bits) | Name | Master | Slave |
|---|---|---|---|---|---|
| - | 0h | - | (Set NMT State) | Response to Set NMT State command. Reflects the command. | Response to Set NMT State command. Reflects the command. |
| | | 0h | PRE-OPERATIONAL | | |
| | | 1h | OPERATIONAL | | |
| | | 2h | RESET NODE | | |
| | | 3h | RESET COMMUNICATION | | |
| | | 4h | STOP | | |
| | | 5h - Fh | - | (reserved) | |
| | 1h | - | (Get Node state) | This response reflects the state set in object 1F82h, see 37, of a CANopen node or network (depending on the value of NodeID). | This response reflects the state set in object 1F82h, see 37, of the module's CANopen interface. |
| | | 0h | PRE-OPERATIONAL | | |
| | | 1h | OPERATIONAL | | |
| | | 2h | RESET NODE | | |
| | | 3h | RESET COMMUNICATION | | |
| | | 4h | STOP | | |
| | | 5h | UNKNOWN | | |
| | | 6h | MISSING | | |
| | | 7h - Eh | - | | |
| | | Fh | ERROR | | |
| | 2h | Bit: | (Get COPM general status) | This response requests the CANopen status of the module | |
| | | 0 | CAN_BUS_OFF | Bus off | |
| | | 1 | CAN_ERR_PASV | Error passive | |
| | | 2 | ERR_NG_HB | Node guarding or Heartbeat error | |
| | | 3 | ERR_SYNC | Sync error | |
| | 3h - 6h | (reserved) | | | |
| | 7h | - | (No operation) | Reflects the command | |

a. The most significant bit in byte 0 is a toggle bit, that is toggled by the module to mirror the toggle bit of the control word.

### 6.2.3 Example

The example shows two control words from the primary network master to the module. Each control word includes a command that affects the secondary CANopen network. Each control word is acknowledged by a status word, that contains a response to the command. Note that the first bit in the control word is toggled when a new command is sent, to make sure it is distinguished from the previous command.




(*) The communication is performed in an ongoing ping-pong fashion. The same command is sent repeatedly, but as long as the toggle bit is not changed, the module will ignore the message. A new command is signalled by the toggle bit changing value.

## 6.2.4 PDO Functionality

Real-time data transfer is performed by means of PDOs (Process Data Objects). The PDOs are linked to entries in the Device Object Dictionary and provide the interface to the application objects. The number and length of PDOs in a device are node specific and have to be configured by the CANopen configuration tool.

PDOs are used both for data transmission and reception, using so called Transmit-PDOs (TPDOs) and Receive-PDOs (RPDOs). Each PDO corresponds to two entries in the Device Object Dictionary. The PDO parameter object holds information on the COB-ID, the transmission type etc. On recognition of the COB-ID the corresponding PDO mapping object can be identified, to make it possible to transmit/receive data to/from the correct object in the device. The default settings for the mapping can be changed during configuration.

### Default PDO Mapping Scheme

The module features a simple default mapping scheme with 4 TPDOs and 4 RPDOs.

- **RPDO**

| RPDO no. | Default COB IDs | Mapped to... | Relating to... | Default State |
|---|---|---|---|---|
| 1 | 200h + Node ID | Object index 2100h, subindex 1... 8 | Receive bytes 2... 9 | Enabled |
| 2 | 300h + Node ID | Object index 2100h, subindex 9... 16 | Receive bytes 10...17 | |
| 3 | 400h + Node ID | Object index 2100h, subindex 17... 24 | Receive bytes 18... 25 | |
| 4 | 500h + Node ID | Object index 2100h, subindex 25... 32 | Receive bytes 26... 33 | |
| 5 | 80000000h | Object index 2100h, subindex 33... 40 | Receive bytes 34... 41 | Disabled |
| ... | | ... | ... | |
| 128 | | Object index 2103h, subindex 121 ... 126 | Receive bytes 506... 511 | |

- **TPDO**

| TPDO no. | Default COB IDs | Mapped to... | Relating to... | Default State |
|---|---|---|---|---|
| 1 | 180h + Node ID | Object index 2000h, subindex 1... 8 | Transmit bytes 2... 9 | Enabled |
| 2 | 280h + Node ID | Object index 2000h, subindex 9... 16 | Transmit bytes 10... 17 | |
| 3 | 380h + Node ID | Object index 2000h, subindex 17... 24 | Transmit bytes 18... 25 | |
| 4 | 480h + Node ID | Object index 2000h, subindex 25... 32 | Transmit bytes 26... 33 | |
| 5 | 80000000h | Object index 2000h, subindex 33... 40 | Transmit bytes 34... 41 | Disabled |
| ... | | ... | ... | |
| 128 | | Object index 2003h, subindex 121 ... 126 | Transmit bytes 506... 511 | |

For more information on the mapping see “Vendor Specific Objects” on page 39

### RPDO Transmission Types

The RPDOs can be received either in synchronous or asynchronous mode. A synchronization (SYNC) object is transmitted periodically by a synchronization master. The data in synchronous RPDOs are not transferred to the application until after the next SYNC object is received. Asynchronous RPDOs will be transferred directly.

The transmission type parameter of a RPDO specifies the triggering mode.

| Transmission type, RPDO | Mode | RPDO transmission description |
|---|---|---|
| 0 - 240 | Synchronous | A received RPDO is transferred to the application after a SYNC object is received. |
| 241 - 253 | - | Reserved |
| 254 - 255 (Default = 255) | Event driven | An RPDO is transmitted without any relation to the SYNC object. |

### TPDO Transmission Types

The TPDOs can be transmitted either in synchronous or asynchronous mode. A synchronization (SYNC) object is transmitted periodically by a synchronization master. Synchronous TPDOs are transmitted within a predefined time-window immediately after a configured number of SYNC objects, or after the SYNC object that follows upon a CoS (Change of State event). Asynchronous TPDOs can be transmitted at any time, triggered by a CoS or a cyclic period set in the Event Timer.

The transmission type parameter of a TPDO specifies the transmission mode as well as the triggering mode.

| Transmission type, TPDO | Mode | TPDO transmission description |
|---|---|---|
| 0 | Synchronous, acyclic | A TPDO is triggered by an event, but not transmitted before the occurrence of a SYNC object. |
| 1 - 240 | Synchronous, cyclic | A TPDO is transmitted with every n-th SYNC object, where n is a defined number from 1 - 240. |
| 241 - 253 | - | Reserved |
| 254 - 255 (Default = 255) | Event driven | A TPDO is transmitted without any relation to the SYNC object. The transmission is triggered by a CoS event or if a specified time has elapsed without an event. |

## 6.3 LSS Services

LSS master functionality according to the CANopen DS305 specification is supported by the module. The module can configure baud rate and node ID of all slaves that support LSS (i.e. the preconfigured baud rate and node ID of a slave can be changed by a master). The module can not act as an LSS slave.

An LSS Slave is identified by its LSS address, that consists of Vendor ID, Product Code, Revision Number and Serial Number of the LSS slave module. If there is a missing slave on the network after the boot timeout, the master will initiate the LSS routine, see Network Management Object "Boot Time, 1F89h" on page 39. It will send an identify slave request, using the LSS address of the slave. If one (and only one) slave responds to this request, the master will set the NodeID on that node to the first missing NodeID. It will then send a bootup message to the node.

# 6.4 Error Control

It is strongly recommended to monitor the network. The Anybus X-gateway CANopen can use either Heartbeat or Node Guarding. At an error event from any of these, the active I/O data is frozen, as no new data will be available.

## 6.4.1 Heartbeat Mechanism

The Heartbeat Mechanism is used to monitor the nodes in the network and verifies that the nodes are available. A heartbeat producer periodically sends a message. The data part of the frame contains a byte indicating the node status. The heartbeat consumer reads these messages. If a message fails to arrive within a certain time limit (defined in the object directory of the devices, objects 1016h and 1017h, 32), a heartbeat event is registered by the consumer. The ERROR LED on the front of the Anybus X-gateway CANopen and the status word will indicate the event. An EMCY object (8130h) is also transmitted on the CANopen fieldbus. If the module is configured as a slave and is in OPERATIONAL state, it will go to PRE-OPERATIONAL state and wait for the user to take action. If it is in master mode, it will take action according to the settings in the master objects.

The Anybus X-gateway CANopen can act both as heartbeat consumer and as heartbeat producer simultaneously.

## 6.4.2 Node Guarding

The NMT Master transmits guarding requests. If an NMT Slave has not responded within a defined time span (node lifetime) or if the communication status of the slave has changed, the master takes appropriate action according to its configuration.

If Life guarding (the slave guards the master) is supported, the slave uses the guard time and lifetime factor from its Object Dictionary to determine the node lifetime. If the slave does not receive a guarding request within its lifetime, a node guard event is registered. The ERROR LED on the front of the Anybus X-gateway CANopen will indicate the event. An EMCY object (8130h) is also transmitted on the CANopen fieldbus.

If the guard time or the lifetime factor are 0 (default), the Slave does not guard the Master. The guarding can be initiated at boot-up or later.

**Note**: The NMT master can monitor a slave either by heartbeat or by node guarding. Only one of these mechanisms at a time can be active. Heartbeat is preferred and if heartbeat is enabled in a slave, any node guarding for that slave is disabled.

## 6.4.3 Emergency Object (EMCY)

The Emergency Object is used for error reporting on the CANopen network when a fatal fault has occurred. The error codes are saved in a list in the Communication Profile Object 1003h, see page 31, and a message is produced on the CANopen network. A list of emergency error codes, that can be produced by the module, is available in "CANopen Emergency Codes" on page 47.

# 7. CANopen Supported Objects

The following sections describe the CANopen objects (secondary CANopen network), according to DS301 and DS302, implemented within the module and described in the EDS file.

## 7.1 Static Data Types

The Static Data Types are implemented according to the DS321 specification from CiA (CAN in Automation).

## 7.2 Communication Profile Area

### 7.2.1 DS301 Communication Profile Objects

The table below shows the objects according to CANopen specification DS301 rev. 4.2.

| Index | Object Name | Subindex | Description | Type | Access | Notes |
|---|---|---|---|---|---|---|
| 1000h | Device Type | 00h | Type of device | U32 | RO | 0000 0000h (No profile) |
| 1001h | Error register | 00h | Error register, connected to the EMCY object. Bit 0 indicates a generic error | U8 | RO | - |
| 1003h | Predefined error field | 00h | Number of errors. Writing a 0 to this subindex clears the error list. | U8 | RW | See “CANopen Emergency Codes” on page 47 for emergency error codes. |
| | | 01h...10h | List of errors. Most recent error at top of list. | U32 | RO | |
| 1005h | COB-ID Sync | 00h | ID of the sync message | U32 | RW | - |
| 1006h | Communication Cycle Period | 00h | Communication cycle period | U32 | RW | Only available if SYNC support is enabled |
| 1007h | Synchronous Window Length | 00h | Synchronous Window Length | U32 | RW | Only available if SYNC support is enabled |
| 1008h | Manufacturer device name | 00h | The name of the CANopen module | Visible string | RO | “Anybus X-gateway CANopen” |
| 1009h | Manufacturer hardware version | 00h | Manufacturer hardware version | Visible string | RO | Current hardware revision |
| 100Ah | Manufacturer software version | 00h | Manufacturer software version | Visible string | RO | Set by HMS |
| 100Ch | Guard time | 00h | Used together with “Life time factor” to decide the node lifetime in ms | U16 | RW | 0000h (default) |
| 100Dh | Life time factor | 00h | If the node has not been guarded within its lifetime (“Life time factor”*”Guard time”), an error event is logged and a remote node error is indicated | U8 | RW | 00h (default) |

| Index | Object Name | Subindex | Description | Type | Access | Notes |
|---|---|---|---|---|---|---|
| 1010h | Store Parameters | 00h | Largest subindex supported | U8 | RO | 01h |
| | | 01h | Store all parameters | U32 | RW | To save a configuration, write “save” = 73 61 76 65h to this object.[a] See also “General Fieldbus Parameters” on page 43. |
| 1011h | Restore Parameters | 00h | Largest sub index supported | U8 | RO | 01h |
| | | 01h | Restore all parameters | U32 | RW | To restore the default values of a configuration, write “load” = 6C 6F 6 1 64h to this object.[a] |
| 1014h | COB-ID EMCY | 00h | Defines the COB-ID of the Emergency Object | U32 | RO | |
| 1016h | Consumer Heartbeat Time | 00h | Largest subindex supported | U8 | RO | 7Fh |
| | | 01h - 80h | The consumer heartbeat time defines the expected heartbeat cycle time and has to be higher than the corresponding producer heartbeat time. Monitoring starts after the reception of the first heartbeat. Not used if 0 | U32 | RW | Node ID + Heartbeat Time.<br>Bits 31-24: reserved<br>Bits 23-16: Node ID<br>Bits 15-0: Heartbeat Time<br><br>Value must be a multiple of 1 ms.<br>Up to 127 nodes can be monitored. |
| 1017h | Producer Heartbeat Time | 00h | Defines the cycle time of the heartbeat. Not used if 0 | U16 | RW | The time must be at least 10 ms and a multiple of 1 ms |
| 1018h | Identity object | 00h | Number of entries | U8 | RO | 04h |
| | | 01h | Vendor ID | U32 | RO | 1Bh (HMS Industrial Networks) |
| | | 02h | Product Code | U32 | RO | 18h (Anybus X-gateway CANopen) |
| | | 03h | Revision Number | U32 | RO | Current software revision |
| | | 04h | Serial Number | U32 | RO | HMS serial number |
| 1029h | Error behavior object | 00h | Number of entries | U8 | RO | |
| | | 01h | Communication error | U8 | RO | 00h: Change to Preoperational if currently in NMT state Operational |
| | | 02h | Profile or manufacturer specific error | U8 | RO | 00h: Change to Preoperational if currently in NMT state Operational |
| 1400h ... 147Fh | Receive PDO parameter | 00h | Largest subindex supported | U8 | RO | 02h |
| | | 01h | COB-ID used by PDO | U32 | RW | - |
| | | 02h | Transmission type | U8 | RW | See “RPDO Transmission Types” on page 29 |
| 1600h ... 167Fh | Receive PDO mapping | 00h | No. of mapped application objects in PDO | U8 | RW | - |
| | | 01h | Mapped object #1 | U32 | RW | - |
| | | 02h | Mapped object #2 | U32 | RW | - |
| | | 03h | Mapped object #3 | U32 | RW | - |
| | | 04h | Mapped object #4 | U32 | RW | - |
| | | 05h | Mapped object #5 | U32 | RW | - |
| | | 06h | Mapped object #6 | U32 | RW | - |
| | | 07h | Mapped object #7 | U32 | RW | - |
| | | 08h | Mapped object #8 | U32 | RW | - |

| Index | Object Name | Subindex | Description | Type | Access | Notes |
|---|---|---|---|---|---|---|
| 1800h ... 187Fh | Transmit PDO parameter | 00h | Largest subindex supported | U8 | RO | 05h |
| | | 01h | COB-ID used by PDO | U32 | RW | - |
| | | 02h | Transmission type | U8 | RW | See “TPDO Transmission Types” on page 29 |
| | | 03h | Inhibit time | U16 | RW | In steps of 0.1 ms |
| | | 05h | Event Timer (ms) | U16 | RW | - |
| 1A00h ... 1A7Fh | Transmit PDO mapping | 00h | No. of mapped application objects in PDO | U8 | RW | - |
| | | 01h | Mapped object #1 | U32 | RW | - |
| | | 02h | Mapped object #2 | U32 | RW | - |
| | | 03h | Mapped object #3 | U32 | RW | - |
| | | 04h | Mapped object #4 | U32 | RW | - |
| | | 05h | Mapped object #5 | U32 | RW | - |
| | | 06h | Mapped object #6 | U32 | RW | - |
| | | 07h | Mapped object #7 | U32 | RW | - |
| | | 08h | Mapped object #8 | U32 | RW | - |

a. Depending on the method of writing to this object, e.g. using a CANopen dongle, the byte order may have to be changed to adapt to the way data is transported on CANopen.

## 7.2.2 Configuration Manager

DS302 part 3: Configuration and program download

### Network Configuration Objects

| Index | Object Name | Subindex | Description | Type | Access |
|---|---|---|---|---|---|
| 1F22h | Concise DCF | - | The concise/compressed DCF files information is stored in this object. | Domain | RW |
| 1F25h | Configure Slave | 0 - 128 | Subindex 0 is ignored.<br>Subindex i (i = 1 - 127): Request reconfiguration of slave with Node ID equal to subindex i.<br>Subindex 128: Request to reconfigure all slaves. | U32[a] | Sub 0: RO<br>Sub 1 - 128: WO |

a. To configure the slave with Node ID i, write “conf” = 63 6F 6E 66h to this object (1F25h, subindex i). If this fails, an emergency code is produced (6161h, see “CANopen Emergency Codes” on page 47).

### Check Configuration

The Configuration Manager (CMT) compares signature and configuration with the value from the DCF to decide if a reconfiguration is to be performed or not. The comparison values are stored by the Configuration Manager in these objects:

| Index | Object Name | Subindex | Description | Type | Access |
|---|---|---|---|---|---|
| 1F26h | Expected Configuration Date | 0 - 127 | The date that the Configuration Manager expects to find when comparing signature and configuration. | UNIT32 | RW |
| 1F27h | Expected Configuration Time | 0 - 127 | The time that the Configuration Manager expects to find when comparing signature and configuration. | UINT32 | RW |

### 7.2.3 Network Management Objects

The NMT master controls the states of the connected network participants, the NMT slaves. It monitors the devices and reports to the application, for example if an NMT slave fails. Please refer to the CANopen specification, see "Related Documents" on page 1. In more complex systems several devices are able to perform as master, which means that the configuration must have an entry defining which device will act as master.

Once configured, the objects carry all information needed for the module to act on the network and the application does not need to be accessed to obtain this information. This results in a substantial reduction of the overall implementation and maintenance effort when implementing multiple applications.

| Index | Object Name | Subindex | Description | Type | Access |
|---|---|---|---|---|---|
| 1F80h | NMT Start-up | - | Defining whether the device is the NMT Master | U32 | RW |
| 1F81h | Slave Assignment | ARRAY | Module list: Entry of all slaves to be managed, including guarding values and the entry of actions to be taken in event of guarding errors. | U32 | Sub 0: RO<br>Sub 1 - 127: RW |
| 1F82h | Request NMT | ARRAY | Remote control initiation of NMT services. For example, tools can use this to request intentional start/stop of individual slaves. Remote query of the current state. | U8 | Sub 0: RO<br>Sub 1 - 127: RW<br>Sub 128: WO |
| 1F83h | Request Guarding | ARRAY | Remote control start/stop of guarding. Remote query of the current state | U8 | Sub 0: RO<br>Sub 1 - 127: RW<br>Sub 128: WO |
| 1F84h | Device Type Identification | ARRAY | Expected device types for the slaves | U32 | Sub 0: RO<br>Sub 1 - 127: RW |
| 1F85h | Vendor Identification | ARRAY | Vendor identifications for the slaves | U32 | Sub 0: RO<br>Sub 1 - 127: RW |
| 1F86h | Product Code | ARRAY | Product codes for the slaves | U32 | Sub 0: RO<br>Sub 1 - 127: RW |
| 1F87h | Revision Number | ARRAY | Revision numbers for the slaves | U32 | Sub 0: RO<br>Sub 1 - 127: RW |
| 1F88h | Serial Number | ARRAY | Expected serial numbers for the slaves | U32 | Sub 0: RO<br>Sub 1 - 127: RW |
| 1F89h | Boot Time | VAR | The maximum time between the start of the boot process and the signalling of successful boot of all mandatory NMT slaves. After this time LSS services are initiated. | U32 | RW |

### NMT Start-up, 1F80h

If a device is to be set up as NMT Master, the master functionality must be enabled in this object. It configures the start-up behavior of the device, and how it will manage the slaves.

**Note**: The Anybus X-gateway CANopen starts up as a slave (bit 0 = 0). For the module to perform as a master, change the value of this bit during configuration and save it to non volatile flash memory by issuing "save" command to subindex 01h in object 1010 (Store Parameters). The setting will take immediate effect, but if not saved, it will be lost at reset or repower.

| Bit No. | Value | Description | Notes |
|---|---|---|---|
| 0 | 0 | NMT Master functionality is disabled. Ignore the rest of the object, except for bits 1 and 3. Ignore object 1F81h. | Default |
| | 1 | NMT Master functionality is enabled. The device is Master | |
| 1 | 0 | Start only explicitly assigned slaves (if bit 3 = 0) | Default |
| | 1 | After boot-up, perform the service NMT Start Remote Node All Nodes (if bit 3 = 0) | |
| 2 | 0 | Automatically enter Operational state | Default |
| | 1 | Do not enter Operational state automatically. Application will decide when to enter Operational state | |
| 3 | 0 | Start-up of slaves allowed (i.e. allowed to send NMT Start Remote Node command) | Default |
| | 1 | Not allowed to send NMT Start Remote Node command. The application will start the slaves | |
| 4 | 0 | If a mandatory slave generates an Error Control Event, treat the slave individually | If bit 6 = 1, ignore bit 4<br>If object 1F81h, bit 3 = 1, the network must not be restarted, if a mandatory slave could not be contacted. |
| | 1 | If a mandatory slave generates an Error Control Event, perform NMT Reset All Nodes (including self) | |
| 5 | - | Not implemented | |
| 6 | 0 | If a mandatory slave generates an Error Control Event, treat the slave according to bit 4 | |
| | 1 | If a mandatory slave generates an Error Control Event, send NMT Stop All Nodes (including self). Ignore bit 4 | |
| 7 - 31 | - | Reserved (0) | |

### Slave Assignment, 1F81h

This object defines which slaves the Master should monitor, control and/or configure. One entry is made for each assigned slave, with the subindex corresponding to the slave's Node ID.

| Bit No | Value | Description |
|---|---|---|
| 0 | 0 | Node with this ID is not a slave |
| | 1 | Node with this ID is a slave. After configuration the node will be set to Operational |
| 1 | - | Reserved |
| 2 | 0 | On an Error Control Event or on detection of a new slave, inform the application, but do NOT configure and start the slave |
| | 1 | On an Error Control Event or on detection of a new slave, inform the application and start the process "Start Boot Slave" |
| 3 | 0 | Optional slave. The network may be started even if this node could not be contacted. |
| | 1 | Mandatory slave. The network must not be started if this node could not be contacted during the boot slave process |
| 4 | - | Not implemented |
| 5 | - | Not implemented |
| 6 | - | Not implemented |
| 7 | 0 | CANopen device may be used without reset to default |
| | 1 | CANopen device shall be reset to factory defaults by issuing a restore to defaults (object 1011h). |
| 8 - 15 | - | 8 bit value for the RetryFactor |
| 16 - 31 | - | 16 bit value for the GuardTime<br>If a slave does not answer, the master will retry the request RetryFactor-1 times with an interval of GuardTime. Guarding will be performed only if non zero values are entered for Retry Factor and GuardTime. |

### Request NMT, 1F82h

Each node on the CANopen network can be controlled individually from the fieldbus application by sending this object. The subindex indicates what nodes the request affects:

| Subindex | Description |
|---|---|
| 0 | Largest subindex supported (128) |
| i (with i = 1...127) | Request NMT Service for the slave with Node ID i. |
| 128 | Request NMT Service for all nodes |

The desired state is given as a numeric value when writing to or reading from the local object dictionary:

| Value | Write access | Read Access |
|---|---|---|
| 0 | - | NMT state unknown. The node is not configured and/or no node monitoring is activated. |
| 1 | - | CANopen device is missing. The node with this Node ID is configured but the monitoring of the node has failed. |
| 4 | STOP remote node | NMT state STOPPED |
| 5 | START remote node | NMT state OPERATIONAL |
| 6 | RESET NODE | - |
| 7 | RESET COMMUNICATION | - |
| 127 | Enter PRE-OPERATIONAL | NMT state PRE-OPERATIONAL |

The entire network can be started with one command (subindex 128)

### Examples

- Node 5 should be transferred to the OPERATIONAL state:
  An SDO write access with the value 5 is executed to object 1F82h subindex 5 in the local object dictionary. When an NMT command is sent, data is cleared.
- All the nodes in the network should be transferred to the PRE-OPERATIONAL state:
  An SDO write access with the value 127 is executed to object 1F82h subindex 128 in the local object dictionary.

### Request Guarding, 1F83h

Guarding can be initiated from the object dictionary in a similar way. Guarding is initiated with the values stored in "Slave Assignment, 1F81h" on page 36, provided that at the same time no parameters are entered for that node as a Heartbeat Consumer

**Note**: This functionality is only supported in master mode.

| Subindex | Description | Access |
|---|---|---|
| 0 | Largest subindex supported (128) | RO |
| i (with i = 1...127) | Request Guarding for the slave with Node ID i | RW |
| 128 | Request Start/Stop Guarding for all nodes. | WO |

| Value | Write access | Read access |
|---|---|---|
| 1 | Start guarding | Slave is guarded |
| 0 | Stop guarding | Slave is not guarded |

### Example:

- Guarding should be started for node 5 (500 ms, Life Time Factor 3):
  An SDO write access with the value 01F40301h is executed to object 1F81h subindex 5 in the local object dictionary. Guarding is activated by an SDO write access with the value 1 to object 1F83h subindex 5 in the local object dictionary.

| Bits | Value | Explanation |
|---|---|---|
| 31 - 16 | 01F4h (500) | The interval with which node 5 will be guarded |
| 15 - 8 | 03h | If node 5 does not answer the guarding will be repeated another RetryFactor -1 times (in this case twice) |
| 7 - 0 | 01h | This value indicates that node 5 is a slave |

### Device Type Identification, 1F84h

Each node on the CANopen network is checked against its expected device type. The subindex indicates which node is checked:

| Subindex | Description |
|---|---|
| 0 | Largest subindex supported (127) |
| i (with i = 1...127) | If the expected device type is not 0 or if the slave is set as mandatory, the module compares expected device type with actual device type (object 1000h, subindex 0) for the slave with Node ID i. If the expected device type is 0, this only gives information about the existence of a node, not which device type it is. If the value is not 0, it is compared to the value read from the node, and boot up of that slave is continued if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL. |

### Vendor Identification, 1F85h

Each node on the CANopen network is checked against its expected vendor. The subindex indicates which node is checked:

| Subindex | Description |
|---|---|
| 0 | Largest subindex supported (127) |
| i (with i = 1...127) | Compares expected vendor with actual vendor (object 1018h, subindex 1) for the slave with Node ID i. Boot up of that slave is continued only if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL. |

### Product Code, 1F86h

Each node on the CANopen network is checked against its expected product code. The subindex indicates which node is checked. The node in question is only checked if data is other than zero:

| Subindex | Description |
|---|---|
| 0 | Largest subindex supported (127) |
| i (with i = 1...127) | Compares expected product code with actual product code (object 1018h, subindex 2) for the slave with Node ID i. Boot up of that slave is continued only if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL. |

### Revision Number, 1F87h

Each node on the CANopen network is checked against its expected revision number. The revision number includes major and minor revision. For a match to occur the major revision has to be exactly the same and the minor revision of the module has to be greater than or equal to the expected minor revision number. The subindex indicates which node is checked. The node in question is only checked if data is other than zero:

| Subindex | Description |
|---|---|
| 0 | Largest subindex supported (127) |
| i (with i = 1...127) | Compares expected revision number with actual revision number (object 1018h, subindex 3) for the slave with Node ID i. Boot up of that slave is continued only if they match according to the description above. |

### Serial Number, 1F88h

Each node on the CANopen network is checked against its expected serial number. The subindex indicates which node is checked. The node in question is only checked if data is other than zero:

| Subindex | Description |
|---|---|
| 0 | Largest subindex supported (127) |
| i (with i = 1...127) | Compares expected serial number with actual serial number (object 1018h, subindex 4) for the slave with Node ID i. Boot up of that slave is continued only if they match. If they don't match, the slave will stay in state PRE-OPERATIONAL. |

### Boot Time, 1F89h

The network master will wait the assigned time (in ms) for all mandatory slaves to boot. If not all mandatory slaves are ready after this time, the LSS routine will be started, see "LSS Services" on page 29. If the assigned time is 0, the master will wait endlessly.

| Value (ms) | Description |
|---|---|
| 0 | Default. No time limit for mandatory slaves to boot |
| > 0 | Time limit for mandatory slave to boot |

## 7.3 Vendor Specific Objects

Vendor specific objects are used to configure the PDOs to the shared memory area. One or several generic data object are connected to each PDO. This is configured during the configuration phase.

Application data bytes 0 and 1, i.e. the first two bytes in the input and output buffers, are used for control and status words.

### 7.3.1 Transmit Buffer

This buffer contains data that is transmitted to the secondary CANopen network.

| Index | Subindex | Type | Access | Name | Position in transmit data area (bytes) |
|---|---|---|---|---|---|
| 2000h | - | STRUCT | | Transmit Byte 1-128 | 2-129 (The first two bytes in the transmit data area are reserved for the Control Word.) |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U8 | RW | Transmit Byte 1 | 2 |
| | 2 | U8 | RW | Transmit Byte 2 | 3 |
| | ... | ... | ... | ... | ,,, |
| | 128 | U8 | RW | Transmit Byte 128 | 129 |
| 2001h | - | STRUCT | | Transmit Byte 129-256 | 130-257 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U8 | RW | Transmit Byte 129 | 130 |
| | 2 | U8 | RW | Transmit Byte 130 | 131 |
| | ... | ... | ... | ... | ... |
| | 128 | U8 | RW | Transmit Byte 256 | 257 |
| 2002h | - | STRUCT | | Transmit Byte 257-384 | 258-385 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U8 | RW | Transmit Byte 257 | 258 |
| | 2 | U8 | RW | Transmit Byte 258 | 259 |
| | ... | ... | ... | ... | ... |
| | 128 | U8 | RW | Transmit Byte 384 | 385 |
| 2003h | - | STRUCT | | Transmit Byte 385-510 | 386-511 |
| | 0 | U8 | RO | Number of entries (value=126) | |
| | 1 | U8 | RW | Transmit Byte 385 | 386 |
| | 2 | U8 | RW | Transmit Byte 386 | 387 |
| | ... | ... | ... | ... | ... |
| | 126 | U8 | RW | Transmit Byte 510 | 511 |
| 2010h | - | STRUCT | | Transmit Word 1-128 | 2-257 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U16 | RW | Transmit Word 1 | 2-3 |
| | 2 | U16 | RW | Transmit Word 2 | 4-5 |
| | ... | ... | ... | ... | ... |
| | 128 | U16 | RW | Transmit Word 128 | 256-257 |
| 2011h | - | STRUCT | | Transmit Word 129-255 area | 258-511 |
| | 0 | U8 | RO | Number of entries (value=127) | |
| | 1 | U16 | RW | Transmit Word 129 | 258-259 |
| | 2 | U16 | RW | Transmit Word 130 | 260-261 |
| | ... | ... | ... | ... | ... |
| | 127 | U16 | RW | Transmit Word 255 | 510-511 |
| 2020h | - | STRUCT | | Transmit Long 1-128 area | 2-511 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U32 | RW | Transmit Long 1 | 2-5 |
| | 2 | U32 | RW | Transmit Long 2 | 6-9 |
| | ... | ... | ... | ... | ... |
| | 128 | U32 | RW | Transmit Long 128 | 510-511 (the last two bytes are padded with zeroes) |

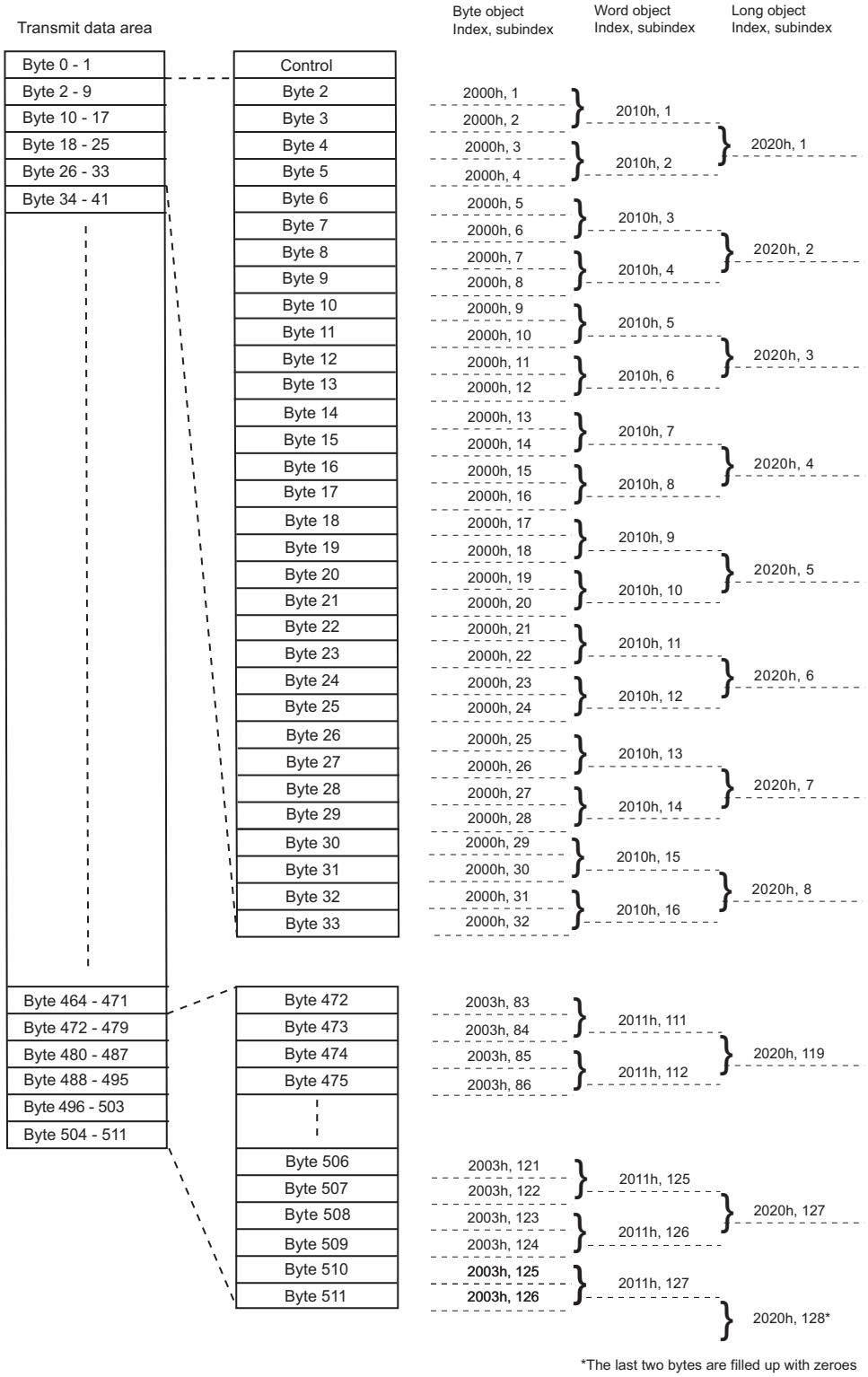
### 7.3.2 Receive Buffer

This buffer contains data that is received from the secondary CANopen network.

| Index | Subindex | Type | Access | Name | Position in receive data area (bytes) |
|---|---|---|---|---|---|
| 2100h | - | STRUCT | | Receive Byte 1-128 area | 2-129 (The first two bytes in the receive data area are reserved for the Status Word.) |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U8 | RW | Receive Byte 1 | 2 |
| | 2 | U8 | RW | Receive Byte2 | 3 |
| | ... | ... | ... | ... | ,,, |
| | 128 | U8 | RW | Receive Byte 128 | 129 |
| 2101h | - | STRUCT | | Receive Byte 129-256 | 130-257 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U8 | RW | Receive Byte 129 | 130 |
| | 2 | U8 | RW | Receive Byte 130 | 131 |
| | ... | ... | ... | ... | ... |
| | 128 | U8 | RW | Receive Byte 256 | 257 |
| 2102h | - | STRUCT | | Receive Byte 257-384 | 258-385 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U8 | RW | Receive Byte 257 | 258 |
| | 2 | U8 | RW | Receive Byte 258 | 259 |
| | ... | ... | ... | ... | ... |
| | 128 | U8 | RW | Receive Byte 384 | 385 |
| 2103h | - | STRUCT | | Receive Byte 385-510 | 386-511 |
| | 0 | U8 | RO | Number of entries (value=126) | |
| | 1 | U8 | RW | Receive Byte 386 | 386 |
| | 2 | U8 | RW | Receive Byte 387 | 387 |
| | ... | ... | ... | ... | ... |
| | 126 | U8 | RW | Receive Byte 511 | 511 |
| 2110h | - | STRUCT | | Receive Word 1-128 | 2-257 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U16 | RW | Receive Word 1 | 2-3 |
| | 2 | U16 | RW | Receive Word 2 | 4-5 |
| | ... | ... | ... | ... | ... |
| | 128 | U16 | RW | Receive Word 128 | 256-257 |
| 2111h | - | STRUCT | | Receive Word 129-255 area | 258-511 |
| | 0 | U8 | RO | Number of entries (value=127) | |
| | 1 | U16 | RW | Receive Word 129 | 258-259 |
| | 2 | U16 | RW | Receive Word 130 | 260-261 |
| | ... | ... | ... | ... | ... |
| | 127 | U16 | RW | Receive Word 255 | 510-511 |
| 2120h | - | STRUCT | | Receive Long 1-128 area | 2-511 |
| | 0 | U8 | RO | Number of entries (value=128) | |
| | 1 | U32 | RW | Receive Long 1 | 2-5 |
| | 2 | U32 | RW | Receive Long 2 | 6-9 |
| | ... | ... | ... | ... | ... |
| | 128 | U32 | RW | Receive Long 128 | 510-511 (the last two bytes are padded with zeroes) |

## 7.3.3 I/O Buffer Addresses and Object Dictionary Indices Relation

Data in the transmit buffer (bytes 2 - 511, from the primary to the secondary CANopen network) are mapped to three different areas in the Local Object Dictionary. The same data is mapped to each area, but in different data types. For example: application data bytes 2 - 5 are mapped to byte object index 2000h, subindex 1 - 4, to word object index 2010h, subindex 1 - 2 and to double-word (long) object index 2020h, subindex 1. Data from the secondary to the primary CANopen network are handled similarly, but with indices starting at 2100h.




*The last two bytes are filled up with zeroes

**Note 1**: The picture shows the Transmit data area. The Receive data area has the same structure, but with indices for byte objects starting at 2100h.

**Note 2:** The first two bytes are occupied by the control/status word, and are used internally by the X-gateway. These bytes should not be used for data exchange. See also "Control Word" on page 24 and "Status Word" on page 25.

### 7.3.4 General Fieldbus Parameters

Index range 3000h-300Fh is allocated for general fieldbus parameters.

| Index | Subindex | Type | Access | Name and Description | Comment |
|---|---|---|---|---|---|
| 3000h | 0 | U16 | RW | Input Data Size (size to EtherNet/IP) | Valid values: 2-512[a], default 16 |
| 3001h | 0 | U16 | RW | Output Data Size (size from EtherNet/IP) | Valid values: 2-512[b], default 16 |

a. The first two bytes of the I/O input area are occupied by the Status Word. The rest is available for data exchange on the secondary CANopen (sub)network side, see "I/O Buffer Addresses and Object Dictionary Indices Relation" on page 42 for further information. Also note that the valid data range may differ depending on the slave interface.

b. The first two bytes of the I/O output area are occupied by the Control Word. The rest is available for data exchange on the secondary CANopen (sub)network side, see "I/O Buffer Addresses and Object Dictionary Indices Relation" on page 42 for further information. Also note that the valid data range may differ depending on the slave interface.

**Note:** Writing to object 1010h (Store Parameters, see 32), will verify the input/output data sizes, stored in these objects, against the current fieldbus limitations. If the data sizes do not comply, an error will be generated (error code 6600h, see "CANopen Emergency Codes" on page 47).

### 7.3.5 EtherNet/IP Specific Parameters

Index range 3070h-307Fh is allocated for EtherNet/IP specific parameters. For this version of the module, they are not used and the index range is reserved for future use.

# A. Technical Specification

## A.1 Protective Earth (PE) Requirements

The product must be connected to protective earth (PE) via the DIN-rail connector in order to achieve proper EMC behavior.

HMS Industrial Networks does not guarantee proper EMC behavior unless these PE requirements are fulfilled.

## A.2 Power Supply

### Supply Voltage

The gateway requires a regulated 24 V ±10 % DC power source.

### Power Consumption

Typical: 100 mA at 24 V.

Maximum: 150 mA at 24 V

## A.3 Environmental Specification

### A.3.1 Temperature

#### Operating

-25° to +55° Celsius
(Test performed according to IEC-60068-2-1 and IEC 60068-2-2.)

#### Non Operating

-40° to +85° Celsius
(Test performed according to IEC-60068-2-1 and IEC 60068-2-2.)

### A.3.2 Relative Humidity

The product is designed for a relative humidity of 5 to 95% non condensing.

Test performed according to IEC 60068-2-30.

## A.4 EMC (CE) Compliance

EMC compliance testing has been conducted according to the Electromagnetic Compatibility Directive 2004/108/EC. For more information please consult the EMC compliance document, see product/support pages for Anybus X-gateway CANopen - EtherNet/IP at www.anybus.com.

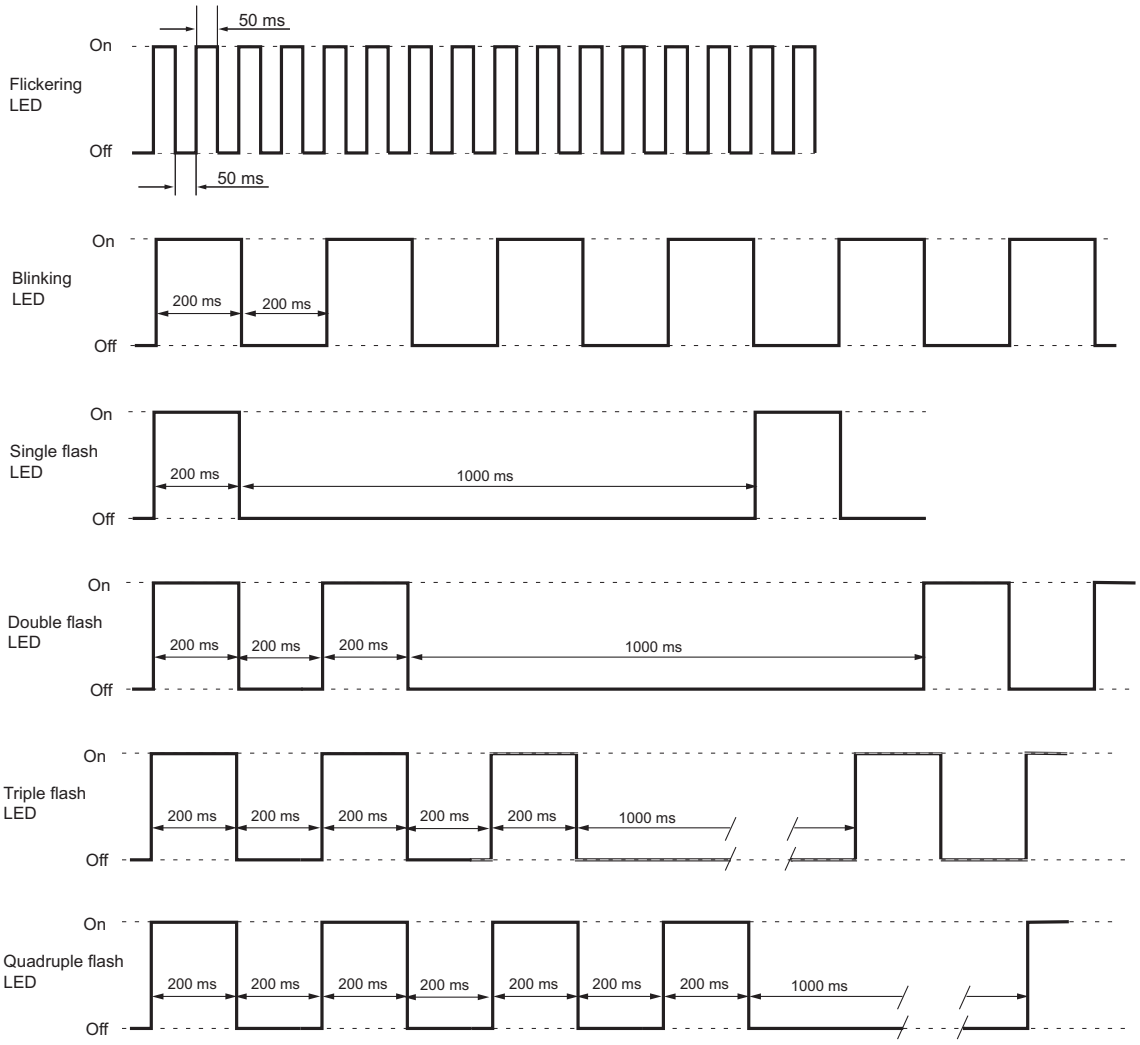
## A.5 UL and ATEX Certification

The Anybus X-gateway CANopen - EtherNet/IP is HazLoc, UL and cUL certified according to file no. E203255. For more information please consult www.ul.com.

ATEX testing has been conducted according to Demko 11 ATEX 1062548. For more information please see product/support pages for Anybus X-gateway CANopen - EtherNet/IP at www.anybus.com.

# B. Status LED Timing Diagrams

The LEDs on the front of the module change their behavior according to the status of the module. This appendix gives the timing diagrams for the different indications, described in "Status LEDs" on page 9.




When LSS services are in progress, both the ERR LED (red) and the RUN LED (green) are flickering.

# C. CANopen Emergency Codes

Below is a list of the CANopen emergency codes that can be produced by the Anybus X-gateway CANopen. The error codes, that have been produced, can be read from the list in the Communication Profile Object at index 1003h, see 31.

| Error Code | Description |
|---|---|
| 0000h | Error reset or no error. |
| 6161h | Software error, only valid in master mode. For additional information, see table below. |
| 6600h | Hardware error |
| 8110h | CAN overrun (objects lost). |
| 8120h | CAN in error passive mode. |
| 8130h | Life guard error or heartbeat error. |
| 8140h | Recovered from bus off. |
| 8210h | PDO not processed due to length error. |
| 8220h | PDO length exceeded. |
| FF10h | (Only valid in slave mode)<br>The control word has been set to no longer allow the module to enter OPERATIONAL state, but the subnetwork is in OPERATIONAL state<br>or<br>a CANopen master attempts to set the module in OPERATIONAL state, while the control word is set not to allow OPERATIONAL state. |

These codes conform to the CANopen standard.

### Software error codes (6161h)

When an emergency code 6161h is produced, additional information is stored in the Communication Profile Object, index 1003h.

| 31 | | 16 15 | 0 |
|---|---|---|---|
| Additional Information | | Emergency error code | |
| Error Code | Node Id (if available) | 61h | 61h |

| Error code | Description |
|---|---|
| 00h | No software error detected. |
| 01h | Tag for CMT record not available. |
| 02h | Cache management inconsistent. |
| 03h | SDO could not be transmitted. |
| 04h | Configuration entry inconsistent. |
| 05h | Check sum error. |
| 06h | Data could not be written to non-volatile memory. |
| 07h | SDO timeout. |
| 08h | SDO error. |

# D. CANopen Configuration Example

This appendix includes an example of how to configure the secondary CANopen network interface of the Anybus X-gateway CANopen using the CANopen Configuration Manager from PORT[1]. Other configuration tools are available and can of course be used instead.

An instruction film with a configuration example is available for download at the support pages at www.anybus.com.

## D.1 Configuration Example

The module is configured as a CANopen slave on the secondary network when the module is started the first time. Default configuration is 16 byte input and 16 byte output data, that can be transferred between the primary EtherNet/IP network and secondary CANopen network. Of these 16 bytes, the first two are reserved for the control/status word. To configure the module as master and change the input and output data sizes, follow the steps below. The next time the module is started, it will start as a master on the secondary network.

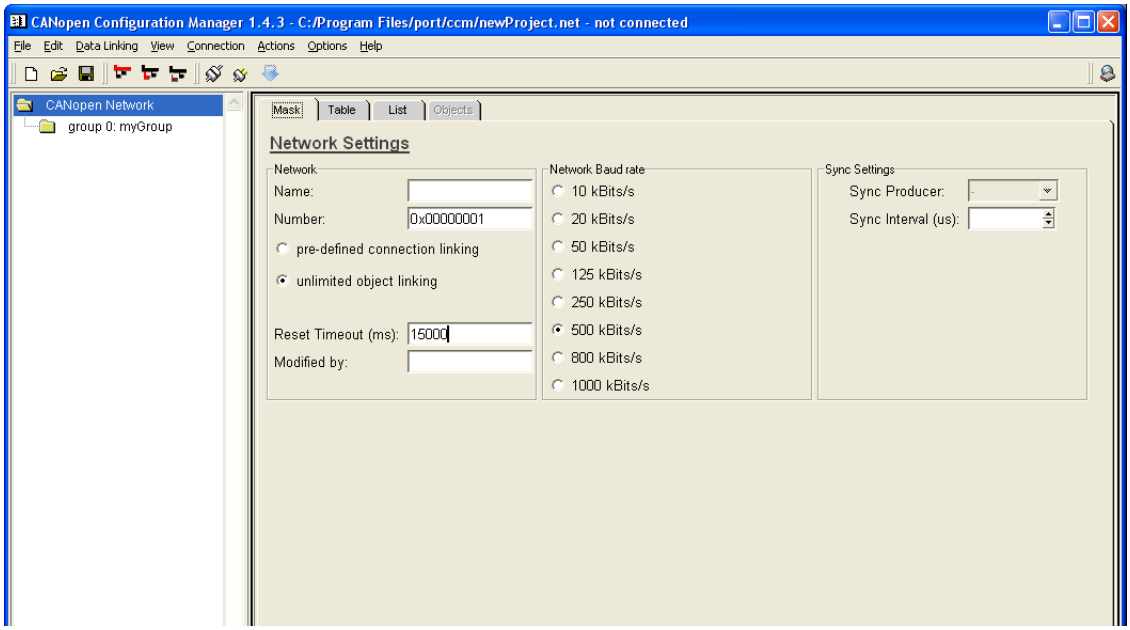
### Step 1. Baud Rate and Node Address

Baud rate and node address are set using the rotary switches on the side of the module, see "Configuration Switches" on page 11.

### Step 2. Set Module in Master Mode

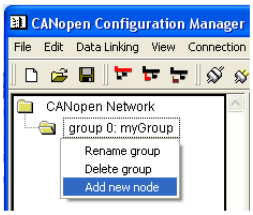
To change to master mode, bit 0 in CANopen object 1F80h (see "NMT Start-up, 1F80h" on page 36) has to be set and saved during CANopen configuration. Using the CANopen Configurator from PORT, this is done as follows:

1. Start the program and select network info. Set the baud rate to match the baud rate set by the configuration switches on the module. Set Sync Interval (if used). Set Reset Timeout to the longest reset time for any node in the network.



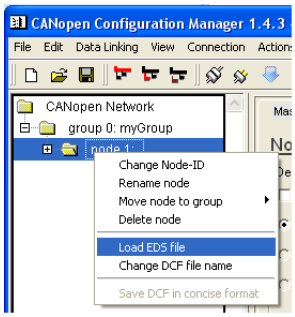

1. Available for download at www.anybus.com. This version can only be used for products from HMS Industrial Networks.

**2.** Right click on group and select “Add new node”.

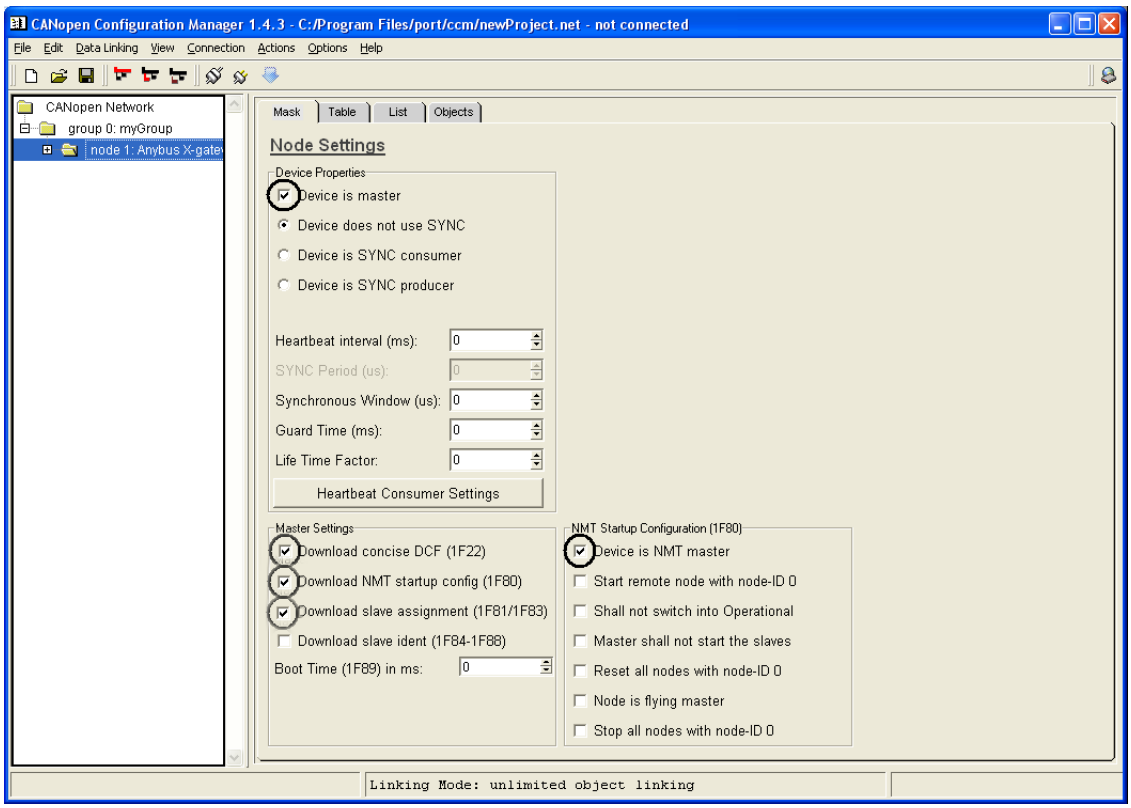



**3.** Open group. Correct node ID can be set for the new node, by right-clicking on the node and selecting “Change Node ID”.

**4.** Right-click on the new node and select “Load EDS file”. Select the EDS file for Anybus X-gateway CANopen “EDS_X_GATEWAY_COPM_V_x_yy.eds” and press OPEN (x_yy for correct version):




**5.** Select tab “Mask”. Then select “Device is master”, which enables you to set “Device is NMT master” in CANopen object 1F80h. For the module to be set properly in master mode, select “Download concise DCF files”, “Download NMT startup config”, and “Download slave assignments” in this view. The module will be in master mode once the configuration is finished.


CANopen Configuration Manager 1.4.3 - C:/Program Files/port/ccm/newProject.net - not connected
File Edit Data Linking View Connection Actions Options Help
CANopen Network
group 0: myGroup
node 1: Anybus X-gate
Mask Table List Objects
Node Settings
Device Properties
Device is master
Device does not use SYNC
Device is SYNC consumer
Device is SYNC producer
Heartbeat interval (ms):
SYNC Period (us):
Synchronous Window (us):
Guard Time (ms):
Life Time Factor:
Heartbeat Consumer Settings
Master Settings
Download concise DCF (1F22)
Download NMT startup config (1F80)
Download slave assignment (1F81/1F83)
Download slave ident (1F84-1F88)
Boot Time (1F89) in ms:
NMT Startup Configuration (1F80)
Device is NMT master
Start remote node with node-ID 0
Shall not switch into Operational
Master shall not start the slaves
Reset all nodes with node-ID 0
Node is flying master
Stop all nodes with node-ID 0
Linking Mode: unlimited object linking

### Step 3. Input and Output Data Size

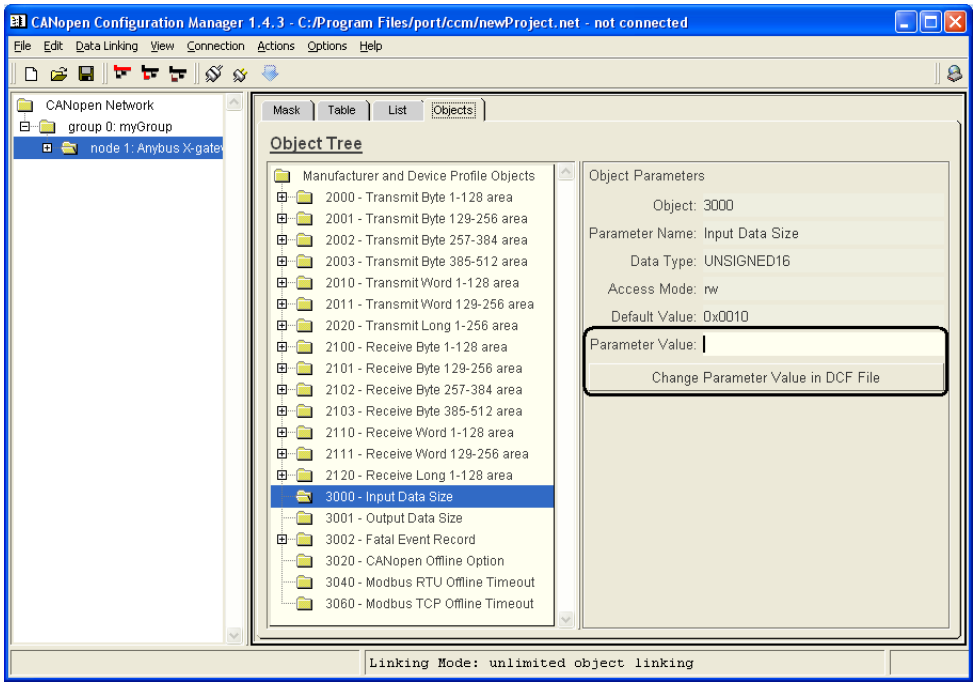
To change the input/output size object 3000h and 3001h have to be modified and saved in the module during CANopen configuration.

The data transferred from the secondary CANopen network to the primary EtherNet/IP network (input), is defined as 2 bytes less than the configured data size in object 3000h. The remaining two bytes are used by the gateway as a status word to indicate status of the gateway to the EtherNet/IP network side.

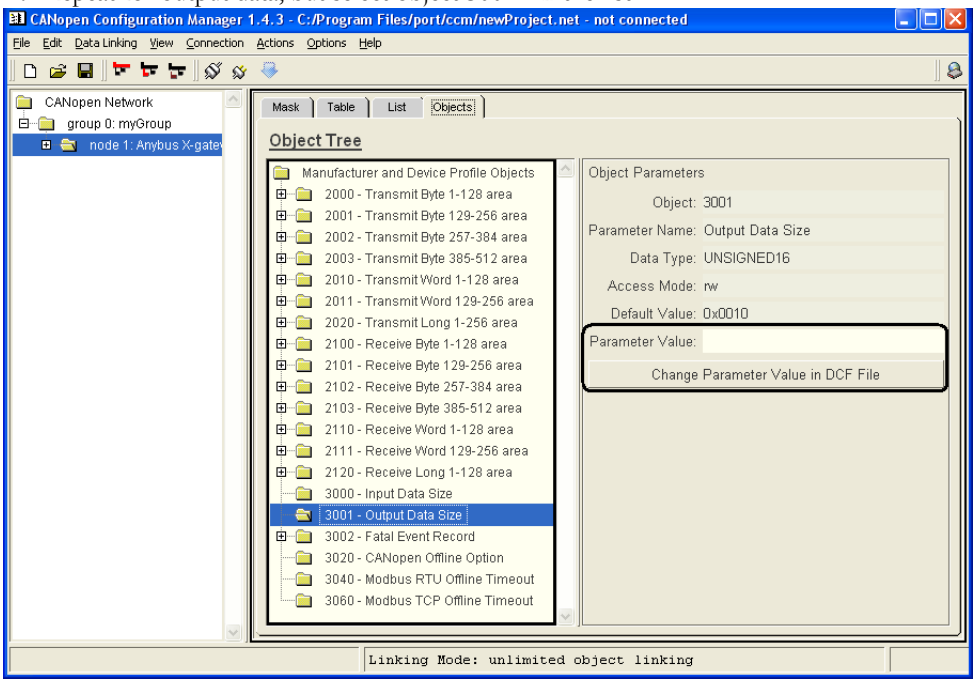
The data transferred from the primary EtherNet/IP network to the secondary CANopen network (output), is defined as 2 bytes less than the configured data size in object 3001h. The remaining two bytes are used for a control word to control the gateway from the primary EtherNet/IP network side.

The input/output size entered is thus the amount of data available on the primary EtherNet/IP network side. This is done in the example below using the CANopen Configurator from PORT. Another configuration tool can of course be used.

**1.** To set Input Size, select tab "Objects" and click on object 3000 in the list. The default value is 10h (16 decimal). Enter the desired value in the box "Parameter Value". If the value is to be stored in the concise DCF file of the module during configuration, press the button "Change Parameter Value in DCF File".


CANopen Configuration Manager 1.4.3 - C:/Program Files/port/ccm/newProject.net - not connected
File Edit Data Linking View Connection Actions Options Help
CANopen Network
group 0: myGroup
node 1: Anybus X-gate
Mask Table List Objects
Object Tree
Manufacturer and Device Profile Objects
2000 - Transmit Byte 1-128 area
2001 - Transmit Byte 129-256 area
2002 - Transmit Byte 257-384 area
2003 - Transmit Byte 385-512 area
2010 - Transmit Word 1-128 area
2011 - Transmit Word 129-256 area
2020 - Transmit Long 1-256 area
2100 - Receive Byte 1-128 area
2101 - Receive Byte 129-256 area
2102 - Receive Byte 257-384 area
2103 - Receive Byte 385-512 area
2110 - Receive Word 1-128 area
2111 - Receive Word 129-256 area
2120 - Receive Long 1-128 area
3000 - Input Data Size
3001 - Output Data Size
3002 - Fatal Event Record
3020 - CANopen Offline Option
3040 - Modbus RTU Offline Timeout
3060 - Modbus TCP Offline Timeout
Object Parameters
Object: 3000
Parameter Name: Input Data Size
Data Type: UNSIGNED16
Access Mode: rw
Default Value: 0x0010
Parameter Value:
Change Parameter Value in DCF File
Linking Mode: unlimited object linking

2. Repeat for output data, but select object 3001h in the list.


CANopen Configuration Manager 1.4.3 - C:/Program Files/port/ccm/newProject.net - not connected
File Edit Data Linking View Connection Actions Options Help
CANopen Network
group 0: myGroup
node 1: Anybus X-gatev
Mask Table List Objects
Object Tree
Manufacturer and Device Profile Objects
2000 - Transmit Byte 1-128 area
2001 - Transmit Byte 129-256 area
2002 - Transmit Byte 257-384 area
2003 - Transmit Byte 385-512 area
2010 - Transmit Word 1-128 area
2011 - Transmit Word 129-256 area
2020 - Transmit Long 1-256 area
2100 - Receive Byte 1-128 area
2101 - Receive Byte 129-256 area
2102 - Receive Byte 257-384 area
2103 - Receive Byte 385-512 area
2110 - Receive Word 1-128 area
2111 - Receive Word 129-256 area
2120 - Receive Long 1-128 area
3000 - Input Data Size
3001 - Output Data Size
3002 - Fatal Event Record
3020 - CANopen Offline Option
3040 - Modbus RTU Offline Timeout
3060 - Modbus TCP Offline Timeout
Object Parameters
Object: 3001
Parameter Name: Output Data Size
Data Type: UNSIGNED16
Access Mode: rw
Default Value: 0x0010
Parameter Value:
Change Parameter Value in DCF File
Linking Mode: unlimited object linking

## Step 4. Add Nodes

Select other nodes and configure them as CANopen slaves. For each node:

- Right click on group and select “Add new node”.
- Set correct node ID by right clicking on the new node under Group and selecting “Change Node ID”.
- Right click on the new node and select “Load EDS file”.
- Select the correct EDS file for the node (as for the master).

Save the project.

## Step 5. Map Data, Generate PDOs and Set the PDO Behavior

When all slaves have been entered, set the CANopen behavior under the tab “Mask” for each node. Then map the data between all nodes under the tab “Table”. When this is done, press F5 for the configurator to generate the PDOs. The PDOs can also be generated by selecting “Generate PDO mapping” from the “Data linking” menu.

Finally, set the behavior of the PDOs under the tab “List” ( parameters “transmission type”, “event timer” and “inhibit timer”.)

### Step 6. Download the Configuration

When all nodes have been configured, save the project again and download the configuration to the network. All modules have to be connected to the CANopen dongle and in PRE-OPERATIONAL state.[1] The configuration is saved in non-volatile flash memory by writing "save" to the CANopen object 1010h (performed by the CANopen Configurator at end of download).

For more information about the CANopen Configurator from PORT, please consult the manual for this product.

**Note**: When the configuration of the secondary CANopen network interface is finished, restart the Anybus X-gateway CANopen module, to ensure that the correct I/O data input and output sizes are set for the primary EtherNet/IP network slave interface.

## D.2 Enabling Data Exchange

To enable the X-gateway to exchange data with the primary network, the PLC controlling the primary network will have to add, initialize and set the fieldbus interface of the module in operational mode.

To start the data exchange with the secondary network, the command "OPERATIONAL" must be sent from the primary network to the secondary network, using the Control Word. If the module is set as slave, this will allow the the module to receive and accept a request from the NMT master of the secondary network to participate in the communication on the secondary network. The module will return an error message if it is not set to OPERATIONAL by the primary network.

If the module is set as master, it will control the operational states of all nodes on the secondary network via the control word.

See also ....

- "Enabling Data Exchange" on page 21
- "Control Word" on page 25
- "Examples" on page 26

1. For the configuration tool, included with CANopen master products from HMS Industrial Networks, to work, the module must be the controlling master in the system, and data linking has to be set up before download.